Browsing a Classification of an Image Collection

Erwan Loisant

UNIVERSITÉ DE NANTES

UFR SCIENCES ET TECHNIQUES

ÉCOLE DOCTORALE

SCIENCES ET TECHNOLOGIES DE L'INFORMATION ET DES MATÉRIAUX

2006

Thèse de Doctorat de l'Université de Nantes

Spécialité : Informatique

Présentée et soutenue publiquement par

Erwan LOISANT

le 10 Décembre 2005

à l'UFR Sciences et Techniques, Université de Nantes

Browsing a Classification of an Image Collection

Jury

Président	:	Noureddine MOUADDIB, Professeur
Rapporteurs	:	Mohand BOUGHANEM, Professeur • Univ. Paul Sabatier (Toulouse)
		Mohand-Said HACID, Professeur • Univ. Claude Bernard Lyon 1
Examinateurs	:	Hiroshi ISHIKAWA, Professeur • Univ. Métropolitaine de Tokyo

Directeur de thèse : Pr. José MARTINEZ

Laboratoire : LABORATOIRE D'INFORMATIQUE DE NANTES ATLANTIQUE. 2, rue de la Houssinière, F-44322 NANTES CEDEX 3

N° ED 366-240

BROWSING A CLASSIFICATION OF AN IMAGE COLLECTION

Parcours de la Classification d'une Collection d'Images

Erwan LOISANT

⋈

favet neptunus eunti

Université de Nantes

Erwan LOISANT

Browsing a Classification of an Image Collection

??+102 p.

Ce document a été préparé avec LaTeX2$_\varepsilon$ et la classe `these-LINA` version 0.92 de l'association de jeunes chercheurs en informatique LOGIN, Université de Nantes. La classe `these-LINA` est disponible à l'adresse :

`http://www.sciences.univ-nantes.fr/info/Login/`

Impression : nantes.tex – 29/6/2006 – 14:35

Révision pour la classe : $Id: these-LINA.cls,v 1.3 2000/11/19 18:30:42 fred Exp

Acknowledgements

The first persons I wish to thank for this work are my two directors, Prof. José MARTINEZ from the Polytechnic School of the University of Nantes on the French hand, and Prof. Hiroshi ISHIKAWA from Tokyo Metropolitan University on the Japanese hand, as well as Prof. Noureddine MOUADDIB from the Polytechnic School of the University of Nantes.

Preparing a Ph.D. between two universities can be both administratively and scientifically difficult, but these three professors offered me their time and energy for my Ph.D. studies to be such a great experience.

Also, I am grateful to the Japanese Ministry of Research and Education (*Monbukagakusho*) that granted me a scholarship while most Japanese students have to fund their studies by themselves.

I acknowledge assistant professors of Tokyo Metropolitan University, Dr. Manabu OHTA and Dr. Kaoru KATAYAMA for their help during my studies. I would also like to thank my Japanese colleagues for the insights that they gave to me: Shohei YOKOYAMA who made me taste every single Japanese dish, and Takuya WATANABE who was my guide into the Japanese *otaku* culture. More generally, all these years I appreciated all the members of Prof. Ishikawa's laboratory, first for welcoming me so kindly despite the poor Japanese level I had when I arrive, then for the numerous karaoke parties on Wednesday night.

Next, let me have a kind thought to all the developers of the open source softwares that I used to implement my work and run the experiments (Ruby, gcc, ImageMagick, LaTeX, Linux, gnuplot…).

Last, but no least, I lasted my family and my friends who were waiting for me in France and encouraged me all along this project. I thank them all for their permanent support.

Contents

Résumé

This chapter is an overview in French of the whole paper.

Les données dites multimédia (images, vidéos) se distinguent des données classique par une densité variable d'information et l'impossibilité de normaliser ces données. Du fait de ces particularités, de nouvelles techniques d'indexation et de recherche d'information ont du être etudiées.

Il y a principalement deux problèmes a résoudre pour la recherche d'information dans les collections multimédia (ou les bases de données multimédia) : (1) la representation des données et (2) le processus de recherche du point de vue de l'utilisateur. Dans le cas des bases de données, l'indexation est fortement liée a ces deux problèmes.

Dans le cas particulier des images, on distingue trois grandes classes:

– la recherche par requêtes formelles, heritée des bases de données classiques
– la recherche avec boucle de retour, où l'utilisateur fait partie intégrante du processus de recherche, et
– la navigation où les images sont organisées en une structure preparée à l'avance, utilisée comme index et comme structure de recherche.

C'est sur cette troisième approche que nos travaux se sont portés ; nous nous sommes en effet interessés au treillis de Galois, une structure de graphe permettant d'organiser les éléments d'une relation binaire.

Une telle structure de navigation a plusieurs avantages sur une approche classique basée sur des requêtes : en particulier, elle permet d'affranchir l'utilisateur d'une phase de rédaction de requête.

Naviguer au sein d'un collection d'images par les treilis de Galois

Dans ce chapitre, nous présentons le meta-modèle de données utilise ainsi que la première proposition de technique de recherche d'images par la navigation.

La representation des données est semi-structurée, et notre proposition est un ensemble de métriques basées sur le contenu de l'image et classées suivant le modèle MPEG-7. Ces métriques sont principalement :

– Des informations de couleurs classées par zones. Les couleurs sont issues d'une segmentation de l'espace a partir du repere HSV (teinte, saturation, luminosité).
– Des informations sur la forme générale de l'image (taille, orientation, élongation)

A partir de ces métriques, à chaque image est associée un ensemble d'attributs formant une relation binaire entre les images et ces attributs. De cette relation binaire est calculé un treillis de Galois, structure utile pour la navigation.

Partionnement complémentaire

Dans ce chapitre, nous proposons d'apporter des réponses au problème majeur posés par les treillis de Galois : le passage á l'échelle.

Pour cela, la structure de navigation est améliorée par un couplage à un système de partionnement, basé sur le projet SAINTETIQ [60] de l'Université de Nantes.

Le treillis n'est alors plus construit sur les images directement mais sur des ensembles d'images similaires. La navigation devient elle aussi à deux niveaux, une navigation *inter-partitions* et une navigation *intra-partition*.

Personalisation et sous-treillis

L'information de contenu des images est généralement insuffisante pour représenter les images telles qu'elles sont vues par un observateur humain. Par exemple, on peut voir une photographie de la Tour Eiffel similaire á une photographie de l'Arc de Triomphe (deux monuments parisiens) tandis qu'un systéme basé uniquement sur le contenu verra plutô l'image de la Tour Eiffel proche d'une image de la Tour de Tokyo (dont la forme est la même).

Cependant, se baser sur une annotation manuelle pose plusieurs problèmes : non seulement cette annotation est très couteuse, mais le résultat est subjectif. Pour une image donnée, deux annotateurs fourniraient un résultat différent ; même un seul annotateur fournirait un résultat différent si on lui demandait d'annoter la même image a quelques semaines d'intervalle.

Dans ce chapitre, nous proposons donc d'offrir à l'utilisateur la possibilité d'établir lui-même les liens entre les images qu'il rencontre. Le problème de la subjectivité devient caduque puisque l'utilisateur effectue l'annotation pour lui-même, et le coût de l'annotation est indolore puisque intégré au processus de recherche lui-même.

Ceci est réalisé par l'application de masques sur une structure commune basée sur l'information de contenu, resultant en une navigation sur des sous-treillis de Galois. Ce procédé est ainsi plus efficace que les systémes de recherche basés sur une boucle de retour, et plus pertinent qu'un systéme basé uniquement sur une structure pré-calculée.

Seuillage dynamique

Dans la partie présentant la navigation par treillis de Galois, nous avons vu qu'un modèle flou était adapté à la representation des images, tandis qu'un treillis de Galois nécessitait une relation binaire entre images et propriétés.

La solution la plus simple, utilisée dans la première partie de cette étude, consiste à appliquer un seuil constant à toutes les images. Cependant, un tel seuil est très sensible au bruit et des nœuds réduits à un élément, rendant la navigation plus complexe font leur apparition.

Dans ce chapitre, nous présentons une technique de seuillage dynamique tenant compte de la structure existante lors de l'insertion de nouveaux éléments. Quand une image doit être insérée, l'insertion dans un nœud existant est préférée á la création de nouveaux nœuds. Pour cela, un seuil plus ou moins sévère est appliqué aux propriétés.

CHAPTER 1

INTRODUCTION

While the ultimate goal of computer science has always been to mimic the human brain [80], in the first days of computing a computer was a machine used to perform mainly mathematical and accounting operations. Shortly, its main goal became to organise data and make the retrieval of information as easy as possible by way of so-called "databases". It is interesting to note that while the English word of "computer" describes it as *a machine to count*, the French word for computer, "ordinateur", means *a machine to sort*, to organise (data). The English word have been decided very soon, while the French word have been decided when the first commercial computers became available for companies, in the 1960's, at a time when databases were already the major applications.

Nowadays, a computer can be used for almost every need (from artistic creation, to playing, to communicating), though retrieving information from larger and larger collections is still dominant. The fact that data sources are now interconnected brought new challenges to the database/information retrieval communities. The time when data was rare and the format could be precisely and strictly determined is over; this century's data is heterogeneous. Retrieval should be done using data of various nature, from various sources, in various format and with various quality.

Moreover, the increasing capacities of the storage devices made it possible even for end-users to keep a large quantity of data, thus enlarging the field of applicability of database and information retrieval techniques. Applications that used to be based simply on a file-system (thus allowing only hierarchical sorting) such as music applications or image viewers are now backed-ended by a database. Consequently a user can now have his or her own *music database* and *photograph database* instead of relying simply on the file-system.

In the meanwhile, the nature of the data to index and retrieve became quite diverse. Originally, structured information appeared as "tables" into relational database management systems (RDBMS), the fields of which were limited to simple types, namely, numbers, strings, and a few extras such as dates. Unstructured information appeared essentially as normalised textual information. Nowadays, it includes structured and semi-structured natural language texts, images, sounds, and videos. To distinguish these new, non-classical data from the former ones, we call them *multimedia data*. Note that if this name suggests that several data types are used together, we follow the convention of the information retrieval community and use this term even if only

one data type - such as images - is used. We use this term as long as the data type on which we work cannot be easily and unambiguously described, because it contains various information, the interpretation of which depends on the observer's culture or sensibility.

Among multimedia data, we are particularly interested in images. In recent years, we saw an explosion of the number and size of digitalised images, not only on the world wide web or on images providers catalogues but also in private collections of individuals. The success of digital cameras made digital images collection accessible to anyone, and recently mobile phones equipped with a camera also participated in the growth of image digital collections. In Japan, from mid-2004 even cheapest devices were equipped with a digital camera.

The images accumulated need to be organised in order to be retrieved easily when the user needs to. Digital camera usually store additional information when recording an image, using the EXIF format: date, flash, focal length... A few of these information, such as date, may be relevant to classify images in a way useful for user. Other informations may also be added to the image when the photograph is taken, for example in mobile phones a positioning system is often available and my be used to locate the photograph in the space. However, these informations are usually not enough to perform a useful classification.

Professional content providers (like Corbis) chose to annotate manually images using a semantic thesaurus. In image hosting websites like Flickr.com, any visitor can annotate the image he or she is visioning by adding any keyword, called within Flickr a *tag*.

The professional annotation approach gives very good results, but very few individuals are willing to take the time to annotate their images for a better classification. On the other hand, the collaborative approach works well for some keywords (like city names or real-world object) but suffers from a lack of standardisation. For example, on the website Flickr a tag such as "cameraphone" is very popular (people who took pictures from their mobile phone) but it has no semantic meaning. A user looking for a photograph of a mobile phone equipped with a camera will be submerged by photographs taken *using* a mobile phone.

Consequently, several authors started to work on a way to organise images in a completely automatic way, using information extracted from the pixels of the image (content information).

Chapter 2 gives an overview of existing projects on content-based image retrieval systems (CBIR). As explained in this chapter, there exists several approaches to retrieve images from an image collection or an image database; the most classical method being *querying* a system where images have been previously indexed.

In our work, we preferred to focus on *navigating* through an image collection that have been organised before-hand in a similarity-based structure. This approach cannot completely replace query-based retrieval, but we believe that it is superior for answering some users' needs. When a user does not have a precise idea of what he or she is looking for, or when his or her idea cannot be easily described by a query (either based on a language or a visual sketch), an approach based on navigation allows him or her to quickly browse a sub-part of the collection in order to rapidly locate the subset of images that s/he is looking for. Being a still visual media, images are particularly adapted to this kind of approach. Effectively, textual documents have to be read, audio files have to be listened to, and video files have to be played back (or a lot of images extracted in order to build a still large storyboard). In contrast, a rather large set of thumbnails of images can be scanned by a human observer in a short time.

Therefore, in this thesis, we introduce a technique for navigating through an image collection using a graph structure useful both for indexing and for navigating, namely a Galois' lattice. A prototype has been developed using some basic metadata on the image content (mostly colour information) presented in section 4. We also present different improvements to this approach, as well as details on implementation in a relational database management system.

1.1 Querying vs. Browsing

The aim of our work is to provide a full proposal for content-based image retrieval, from metadata extraction to image search itself, to be implemented into a fully working prototype. Rather than successively executing query on a database, our work is focused on a navigation-based process that integrates the user into the search process and thus ensures a permanent feedback between the user and the system.

A common problem in CBIR systems is that the user is usually unable to understand the underlying model used by the system. Cases where a user looks for an image knowing which colours or shapes he or she likes, and how to describe it, are very rare. In a navigation process, the user does not need to describe his or her need. He or she just browses the collection, constructing a path from an entry point to the images he or she mostly likes.

When looking at most proposals of navigation through a multimedia collection, we noticed that they are usually built as a layer over similarity search: they recreate a new state from user input, and display it to the user for another interaction.

Kaester *et al.*'s work [34] proposes to combine several input methods to search for images, including touch screen (to select parts of images or perform gestures) and speech recognition. By using these non-classical input methods Kaester could produce a graphical interface that makes the user feels like he or she is navigating the image database, however this system is still based on similarity search: the user will actually select images or parts of the images for the system to find images similar to these samples. The collection is stored by using multi-dimensional indexing techniques.

In their project El Niño, Santini *et al.* worked on integrating browsing and querying [69]. Their proposal is a set of search engines connected by a mediator that dispatches the queries to the search engines, collects the results and displays them to the user. Images are arranged on a two-dimensional plane, and the user interacts with the system mainly by two ways:

- By clicking on an image, the user asks the system to move this image to the centre. From the user's point of view, he or she is moving inside the collection; from the system's point of view, the user is launching the query "find and display the images similar this one."
- By drag-and-dropping images, the user teaches the system similarities that were not present. The user can then tell the system that from his or her point of view, two images are similar. In other words, this is a user-personalisation process.

Our work is very different from Santini's or Kaester's ones in the sense that it is based on a navigation structure that is built before-hand. Consequently, there is no calculation during the search process; this leads to a very fast and responsive system.

Since other systems formulate a query at each iteration, they have to face (1) the cost of the formulation of a query using user feedback and (2) the cost of the execution of this query. In a multidimensional space, both of these operations are quite costly.

Compared to other systems performing navigation through image collections [32] [68], our system has the particularity to use directly the index structure for the navigation itself. On the contrary, the others build during retrieval process a path that is not directly linked to the index structure.

While we have to face a more costly process to index our images, once the collection is indexed, the search process is very fast and responsive. Additionally, it is very easy to publish. The navigation structure can even be produced in the form of a set of static XHTML (see Figure 4.6), and for example burned on a CD-ROM.

1.2 Outline

State of the Art

This first part is a bibliographical work on content-based image retrieval (CBIR). Here we present the two problems of CBIR, the specificities of this branch compared to classical problems of *databases* or *information retrieval*: (1) the data representation and (2) the search process itself from a user point of view.

Usually used techniques are also described in this part:

– *Similarity measures*, an approach even now widely used for CBIR, and
– *feedback querying*, algorithmically more costly than simple similarity-based query but much more relevant. Moreover, feedback querying introduced the idea of integrating user into the search loop.

We then present a more recent approach for CBIR, whose the current proposal is part of: *navigation* through an image collection. In this section, we precise the main advantage of navigation over the other approaches: there is a continuity in the search process; rather than incrementally make queries on the system, the user just navigate through it converging to the image s/he was looking for.

Navigating an Image Collection Using Galois' Lattices

In this chapter, we present the meta-model used as well as our first proposal of navigation through a collection of images.

Data representation is semi-structured, and our proposal is a set of metrics based on image content and classified according the MPEG-7 model. These metrics are mainly:

– Colour information classified by zone. Colours come from a segmentation of the colour space from the HSV space (hue, saturation, value).
– Informations on general shape of image (size, orientation, elongation).

From these metrics, an attribute set is associated to each image forming a binary relationship between the images and these attributes. From this binary relationship a Galois' lattice is calculated. A Galois' lattices is a structure useful for navigation, detailed in section 4.2.3.

During our experimentations, we were able to build a lattice of slightly more than 5,000 images, this number being mainly limited by the space complexity of the algorithm.

As a navigation structure, the advantages of Galois' lattices are really numerous.

– First of all, it is very fast to navigate through a graph structure that has been computed off-line. If we neglect the time required to load sample images, navigating from one node to another is optimal, i.e., in $O(1)$. This was one of the main requirements.
– Then, a Galois' lattice is intrinsically a multi-dimensional classification technique. Indeed, no dimension is privileged. Hence, it can be seen as a structure that dichotomises the hyper-cube associated to the property subsets along any hyper-plane.
– Consequently, the distance from the inf or sup nodes[1] of the graph to any other node is at most logarithmic in the number of used properties.
– Next, this tool is insensitive to correlations. There is no distance computation. If all images with a given property (almost) always exhibit another property, then the images will simply be located within the same node.
– Also, this tool helps to correct the users' mistakes very easily. Whenever a user selects a direct descendant node, he or she adds implicitly a new constraint. If he or she figures out, much later, when seeing more specific sample images, that this browsing direction is slightly bad, he or she just has to move to a different direct ancestor node. This operation removes a constraint and undoes the erroneous move without having to go back to the point where the "error" actually occurred.
– The Galois' lattice structure easily hides unwanted features. This is a problem that cannot always be taken into account by similarity measures. (A counter-example is Surfimage [50], but the measures are limited to mean and variance of supposed Gaussian distributions.)

However, Galois' lattice used as a navigation structure also have drawbacks.

– Constructing a Galois' lattice is not an easy task. The time complexity is in $O(n^2)$ where n is the number of nodes (see the details in the section 4.2). Theoretical improvements on this bound are still unknown to our knowledge, and algorithmic variants do not achieve actual improvements in the implementations [26].
– Also, the description space associated to a Galois' lattice is exponential in the number of properties. (We easily use several hundreds!) Of course, this should not occur, unless we index such a large number of images. However, if several images share common properties but have unique properties too, then a (localised) exponential explosion appears.

The first drawback, scalability, is addressed in section 5 by proposing to navigate on *image clusters* rather than directly on images.

The second drawback is addressed is section 7 with a technique to build the lattice while limiting the creation of new nodes, and avoiding creation of nodes concerning a very small number of elements.

[1] inf and sup are special nodes presented in the section 4.2, where Galois' lattices are detailed.

Complementary Clustering

In the previous section, we claimed that *scalability* is an important issue for Galois' lattice. Indeed, the time complexity is quadratic and the space complexity is exponential. Consequently, depending on the computer speed and memory we can easily build a Galois on 5,000 images; if the data have a lot of similar images, on a system with a lot of memory we can expect to reach 10,000 images. However, even with the increase of computer power and the decrease of memory price, it is very unlikely that we can ever build the Galois' lattice for a bigger collection using current algorithms. This is a serious problem since today's images collections may reach the million of images.

Thus, we propose to improve the structure by associating it to a clustering system, based on the SAINTETIQ project [60] of Nantes University.

The lattice is no longer constructed directly on images but on similar images collections. These collections are supposed to be different from each others and internally homogeneous. Navigation becomes two-level, an *inter-cluster* navigation and an *intra-cluster* navigation.

Using such a technique, if we build about 3,000 clusters of 200 images, we can reach very large databases, *i.e.* about one million images.

Personalisation and Sub-lattices

Content information of images is usually not enough to represent them as seen by a human observer. For example, a photograph of the Eiffel Tower and a photograph of the Arc of Triumph can be seen as very similar: both represent a monument of Paris. However, a system based only on content will lack the information "Paris" and would rather see the Eiffel Tower similar to Tokyo Tower, that have the same shape.

However, using only manual annotation has several drawbacks: not only this annotation is very expensive, but the result is subjective. For a given image two annotators would produce two different results. Even the same annotator would produce a different result if asked to annotate the same image after a few weeks.

In this chapter, we propose to offer to user to establish himself links between images he or she can find. The problem of subjectivity becomes pointless since user establish the annotation for himself; there can be no distortion between the annotator and the user. Moreover, the cost of annotation is painless because it is integrated into the retrieval process.

This is implemented by applying masks on a structure common to all users based on image content, resulting in a navigation on sub-lattices. This process is more efficient than systems using a user feedback for querying, and more accurate than systems based solely on a structure calculated before-hand.

Dynamic Threshold

In the part introducing Galois' lattices, we established that while a fuzzy model was adapted to represent images, a Galois' lattice needs a binary relationship between images and descriptions.

The most trivial answer, used in our first prototype of $Click^{Im}_{AGE}$ [40] and in the first part of this work, was to apply a constant threshold to all images. However, such a threshold is very sensible to noise and nodes reduced to a single element appear, making navigation more complex.

In this part, we present a dynamic threshold technique taking existing structure into account when inserting a new node. When an image is to be inserted, this algorithm will prefer to insert it into an existing node rather than to create a new one. To achieve it, the threshold applied to property will vary according the existing lattice.

PART I

State of the Art

CHAPTER 2

CONTENT-BASED IMAGE RETRIEVAL (CBIR)

2.1 Introduction

Image databases are part of digital libraries. Research performed in the last ten years led to several prototypes: Amore (Advanced Multimedia-oriented Retrieval Engine), BlobWorld, CANDID [35], Chabot/Cypress [52], CORE (Content Object Retrieval Engine) [84], FIRST (Fuzzy Image Retrieval SysTem), $Find^{Im}_{AGE}$ [39], IDQS (Image Database Query System) [83], ImageRover, Jacob, MARS (Multimedia Analysis and Retrieval System) [53], MetaSeek, MIR (Multimedia Indexing and Retrieval), MMIS (Multimedia Information System), MULTOS (MULTimedia Office Server) [45], NeTra [38], Picasso [11], PicHunter, PIQImage, PhotoBook [55], QBIC (Query By Image Content) [19], RetrievalWare, SQUID (Shape Queries Using Image Databases), SurfImage [49], Virage, VisualSEEK [73], WebSEEK [10], WebSeer, Xenomania, etc.

Although the wide use of image databases is quite recent, they have been integrated in most commercial systems (such as Multimedia Manager from IBM, Oracle and SQL Server from Microsoft) for several years. These systems may be used in applications involving images, such as interior design tools or simply retrieval in images collections, from image providers (such as Corbis™), or personal photographs collections (like Google's Picasa™, or Gnome's F-Spot). A recent application is image hosting websites (such as Flickr.com, Fotoflix.com or Buzznet.com) offering to anyone to post photographs, either to share it with family and friend or to publish them publicly.

In the first section, we shall explain the specific problems of image databases (IDB). In short, it is needed to design a description scheme for images based on various informations (colour, shape, format, human annotations...). Then for each property a similarity measure must be described. Finally, various retrieval methods have to be designed to satisfy various users goals. This tend to a very flexible architecture.

Second part will cover most common properties (both content-based and annotation-based) used for IDB, as well as similarity measures on them. Then the three main retrieval techniques

- query, relevance feedback and navigation - will be presented with examples. A particular attention will be given to navigation as the current proposal is a navigation-type retrieval method.

2.2 Generalities on Image Retrieval Systems

Most general case is the open systems case, such as the World Wide Web, where images and users are very various. First, using knowledge on the content type is not possible. Moreover, the user is not an expert and cannot understand most of the properties associated to images and consequently the reason why some images may be returned after a given query. Finally, an open system has a quick turn-over rate: images are often added and removed, limiting the property calculus to low-cost algorithms.

In the contrary, a lot of bases are limited to images of a particular class. These domains are numerous: news images, architecture images, medical images, botanic images, satellite images, face images, etc. Working on a particular class gives additional informations that must be exploited.

2.2.1 Problem definition

The main specificity of images as data is that, unlike for example textual information, the similarity between two images is far to be the direct similarity of low-level information, pixels.

2.2.1.1 Defining Content

The first difficulty of the study is on the definition of the word “content”. Image representation is semi-structured [1] and can be qualified as:

- irregular: several image types coexist (grey-scale, colour, with or without colour index…);
- incomplete: all informations are not necessarily extracted for all images. This is mainly due to the cost of processes and the existence of heterogeneous sources, for example in QBIC [19];
- extensible: one must be able to add new properties to take into account new techniques or integrate a new image type;
- applicative: images are not manipulated independently but rather in a schema. In this domain, authors agree to consider a hierarchy between low-level data and high level data (semantic and logic) from artificial vision proposals [3].

Lower level is the intrinsic content, directly linked to the physical signal and usually represented by a matrix of pixels, the canonic representation. Automatic processes can then extract qualitative and general informations from images, the most classical being histograms.

On the other hand, the external or semantic informations have to be provided manually. The title, subject, author are part of this category. Keywords are the most generic metadata an image can receive; they may be organised in a thesaurus or not. These metadata are usually not quantifiable, but belief degrees may smooth the transition (“very”, “quite”, “a little”). The

problem then becomes the same as textual information retrieval: incomplete index, ambiguity, time and space variable [20].

However, with the increasing digitalisation techniques, some semantic attributes may be associated automatically to the image: author, place, date and time, etc. These are a mix of semantic and quantifiable informations.

In the same way, images associated to documents give additional informations. For example, WebSEEK [10] and Google Images indexing web pages use the tag content. Actually, these systems make use of a manual index that is not done at the moment of the insertion into the database but "at the source".

Moreover, techniques to automate the semantic knowledge extraction exist, applicable to reduced images classes:

- After a learning phase, some keywords may be automatically associated to images regions (sky, grass, leafs, skin...) [62] [56];
- In the domain of Renaissance paintings, [11] exploits rules from an art book to derive even feelings from images. For examples, the religious feeling is represented by an important quantity of red and blue;
- Chabot [52] gives a simple horizon detection algorithm (indexed images being landscape images);
- [78] proposes a classification between indoor pictures and outdoor pictures;
- etc.

Finally, note that an image database, unlike an image retrieval system, gives naturally additional informations. The schema of the database and the application gives the structure of the base and consequently the structure of included images. For example, an image associated to a "City" instance has a high probability to be an urban image.

Of course, the transition from visual content to metadata is progressive. For example, segmentation techniques isolating significant regions from an image require knowledge and may be very specific (adapted to medical images), or very general (for example based on colour).

We can oppose the low-level data and the high-level data on several criteria:

- intrinsic to extrinsic content;
- general to specific properties;
- quantitative to qualitative properties
- automatic to manual extraction;
- objective to subjective properties.

The hierarchy defined causes a major difficulty for retrieval, related to subjectivity. Indeed, the system works on the low level data while the user thinks in the higher level [67].

On the one hand, some authors prefer the minimal semantic extraction. Indeed, experiments show that using semantic informations greatly improves the results quality [52] [9].

On the other hand, other authors prefer an approach where only the low-level data is explicitly used. There are two reasons to this choice: the existence of a *latent semantic*, and the introduction of subjectivity in high-level semantic.

First of all, combining low-level data leads to satisfying results. Then, a latent semantic exist: for example the grass is usually green, the sky is blue and the skin of Caucasian people

is orange. CANDID actually makes use of this approach, where combining several low-level characteristics can make an implicit high-level criteria appear.

Another reason to limit deliberately the highest knowledge level is because, becoming subjective, it does not apply any longer to all users. [67] even argues that since the semantic of an image cannot be formally described, we should limit the study to finding correlation between user's goals and characteristics that can be extracted from images. It is not reasonable to ask a system to see a similarity between a portrait of the French President Chirac and a photograph of the Japanese Prime Minister Koizumi for the only reason that both of them are political leaders. Even from the name it would be a hard task. Consequently an image retrieval system should be limited to a system to find relevant images using vision capacity but not automatic vision. The system cannot "see" the images like an human would do, but will rather help a human to classify them automatically.

2.2.1.2 Defining retrieval

Content being defined, an information retrieval system has to be designed. However, the users are various; for that reason the system should offer several retrieval methods. Actually, representing queries is as important as representing images. More precisely, an image retrieval system is a couple of image representations and query representations.

The following list presents several interrogation ways that have been implemented in several prototypes:

- The oldest is of course queries from keywords describing the image contents, image databases becoming textual databases;
- the interrogation from a sample image quickly appeared, the image being one from the base or provided by user;
- a variant is asking the user to sketch the image he or she is looking for (for example "a red disk on a green background" to find rose images [19]);
- for a better precision, especially in textures, an image may be constructed from fragments of other images from the base [38];
- an extension is an annotated sketch (globally or locally) to precise characteristics that should be taken into account [73] and possibly retrain them to particular values;
- this last proposal is close to query using a formal language combining structural data and visual content [52] [39];
- an extension of this approach is to provide semantic descriptions like "find images including a character in a red dress with, at his feet, a small white dog";
- finally, relevance feedback approaches appeared, system trying to discover or simply improve conditions that will make an images subset relevant.

There are indeed a large panel of possibilities but it can be clarified. We can actually classify retrieval techniques into three basic and complementary approaches, whatever the usage is to be done:

- Formal querying where accent is put on specifying the image to find, in other words the construction of a query (even if the interrogation language may be very simple, reduced to a linear function);

– feedback querying where the user is a part of the retrieval process;
– navigation-based "retrieval" allow the user to move into a before-hand calculated structure.

Formal querying Formal querying may be done via a query language as well as in a graphical way, thus interactive.

Graphical querying is more comfortable and usually simplified form of the query writing [8] [38] [73]. However, it can be almost necessary in some cases, like selecting a colour from a palette rather than describing digital values [19].

Retrieval by formal query actually consists in specifying the description of a virtual image, i.e. a value range that several attributes should respect, these properties being linked by an implicit conjunction.

Despite the apparent simplicity of this "classical" approach, two new elements appear. First, it is better to use an approximate interrogation rather than a strict interrogation. Then, multi-dimensional indexing problems appear.

Using a formal query language brings several advantages. It allows querying an image database with a strong integration of digital data, images contents [52] [39] as well as the application schema. Still in the case of a multimedia DBMS, queries may be done on a subset of the base [9] improving both the process time and the result quality. Queries may also become views allowing to solve other queries [52]. Note that this is the only way to build non-interactive applications.

Feedback querying The previously described formal approach is valid neither for ad hoc interrogation, nor for naive users. Feed-back querying is very important in an image retrieval system: images cannot be easily described. Consequently, redaction "mistakes" are common and require rewriting, until finally obtaining a relevant images subset.

It is necessarily interactive; the user is actually a part of the retrieval process. The result of the query is not one of the subset successively displayed to the screen but only the "last". Finally, the user decides when the retrieval process is finished (unless the system is unable to return any image). For that reason, for the user to perform the retrieval intuitively and efficiently, the user interface should be carefully designed. In this way the user will make better use of his cognitive capacities and improve the results convergence. Of course, the system performances and reactivity will also improve the user experience and thus the retrieval efficiency.

In the general approach, the user will annotate several images as examples or counter-examples. The system must features inference techniques [49] [41]: it will have to determine automatically the relevant criteria. In other words, it will have to discover weights that discriminate the examples from the counter-examples. At each iteration, the system constructs and improves, with the help of the user, a new formal query.

Let's note that the most commonly used approach used to be lookup from a unique sample image, or a particular region from an image [83]. The user's work is then to set weights on different properties. This is between a formal approach and a feedback approach: the final specification is obtained by the "right" weights and the "right" sample image.

Navigation After these two retrieval forms, the third way is the possibility to navigate through the schema of the database (images but also related data in the image database). It has been showed that in certain cases, hypermedia can be an alternative to a DBMS [51].

Navigation is a very efficient retrieval form that, as well as relevance feedback, combines the system advantages with the user's vision. Indeed, if the similarities between images have been stored in the base, find the images that are similar to a given images is done in an instant: the system's work has been done before-hand. Then, the user is to choose an exploration direction rather than another. The navigation technique itself may be more or less complex, but the main work consists in a before-hand classification [9] [29] [70]. This approach being also interactive, it can easily be combined with the previous retrieval modes [32]. The interrogation allows to find entry points in a "clusters" structure, then the navigation allows to explore these sub-spaces.

The classification results may also be used in formal queries, images classes being interpreted as views (subsets) or taxonomies (additional properties) or more simply as binary associations between close images.

2.2.2 Architecture

From these generalities, we note that the points to make clear are: the images representation, the similarity measures, the physical index techniques, the user implication in retrieval process and more recently the images classification.

Thus, the heart of an image retrieval system or an image database must be flexible [84]. First, it must adapt itself to new informations, known before-hand or learnt, that means that the design must be very open. Then, it must discover the user's goals to improve results relevance, including experimented users.

On a technical point of view, chosen architecture should not be a limiting factor. The module should be adaptable as well in database driven applications as in more open systems, like artificial vision systems. However, we will not talk about this aspect.

2.3 Commonly Used Techniques for Image Retrieval Systems

Elements introduced about architecture will now be partially formalised. We should indeed deal with a set of extensible and application-adaptable elements. For that reason, we will limit our study to fundamental elements.

Images will be modelled as semi-structured objects through a set of properties of very various nature. Several mathematical tools are used to characterise one or more property types: mathematical morphology, fractals, statistics, transforms (Fourier, wavelet, Gabora. . .), etc. We will keep statistical tools as simple as histograms as a basis to discuss the choice of good characteristics. Then, we will see how to build similarity measures on these properties as well as on their combinations. A few inference techniques will be cited.

2.3.1 Images representation

While being very important, images representation is very open. Consequently, the schema of an image class in an image retrieval system may only be introduced in a generic way, as a semi-structured object:

$$I = C^{\mathbb{N}} \tag{2.1}$$

Where C is a polymorphic characteristics set. Later, we shall use C_C to talk about the set of values of the C characteristic as well as the relating $I \to C$ function.

It is a hard task to find characteristics that represent the best an image content. These properties have to respect as much as possible the following criteria, that are sometimes opposed [5]:

- exhaustivity: characteristics must cover the whole important elements of image;
- compacity: discriminant information coding must be compact in order to reduce simultaneously storage and process costs, either before-hand by choosing relevant elements, either later by dimensionality reduction techniques [16]
- robustness: characteristics must be tolerant to the noise that appears for different reasons: photograph taken in bad conditions, digitalisation done with poor material, strong image compression...
- discrimination: though characteristics are supposed to be complementary and used together, each characteristic must by itself allow to differentiate a lot of image classes;
- precision: characteristics should be calculated with a precision equal to human eye, in order to allow an advanced discrimination;

The compacity condition prevents the use of canonic representation of images. For instance, Chabot stores only thumbnails in the database, images itself being stored on optical disks [52].

There are several pieces of data to take care of in order to organise adequately an image database [3]. The standardisation effort of MPEG-7 [47, 48] separates:

1. the format information (stereo for audio, infra-red for image...),
2. the physical information (sound energy, main colours...),
3. the perceptual information (male voice, hot colour...),
4. the structural information (splitting a video into planes, an image into regions...),
5. the intrinsic metadata (keywords...),
6. the miscellaneous annotations.

In this study, we will use the MPEG-7 classification to define metadata; however we will not make use of every layer proposed by the MPEG-7 proposal.

2.3.1.1 Low-level data

Colour and textures are one of the most used low-level characteristics, because it is very close to the description that a human observer may do. Several colour models exists, answering to different needs, from physical models (RGB being the main) to perceptual models based on a separation of value, saturation and hue and normalised models (XYZ, L*u*v*, L*a*b*). Texture is generally represented by granularity, contrast and direction [19].

Since [77], histograms are still a base for a large number of propositions:

$$h: \begin{array}{l}(E_j) \to (E \to [0,1]) \\ (e_j)_{j\in J} \mapsto \{e \to \frac{(e_j|e_j=e)_{j\in J}}{(e_j)_{j\in J}} | e \in (e_j)_{j\in J}\}\end{array}$$

Where (E_J) is a data family, indexed by elements of the J set, and $E \to [0,1]$ is the function associating its frequency to each data family, i.e., the corresponding histogram.

Histograms may be single or multi-dimensional. For example, in a quantified colour space, one can create tri-dimensional histograms with $J = \mathbb{N} \times \mathbb{N}$ for the coordinates system and $E = \mathbb{N} \times \mathbb{N} \times \mathbb{N}$ for pixel values.

Histograms feature exhaustivity, robustness and precision. However, they are neither particularly compact nor very discriminant in large IDB [54].

The compacity problem may be solved by creating value classes, also reducing processing time. In the case of colour, one may create conceptual classes, i.e., a non-uniform segmentation of the colour space (9 hues in $Find^{Im}_{AGE}$ [39]). This segmentation may even suppress the tri-dimensionality of the colour space, by replacing the three histograms by a unique one. However statistical measures cannot be done any longer on such histogram.

The discrimination problem is deeper. It is due to the lack of correlation between an histogram's modalities. The compression we discussed above solves it partially but in a too strong way since it limits the number of images classes that can appear.

When using a full histogram is not possible, using an acceptable number of inertia movements (average then centred moments) can express as finely as possible the histogram's shape. Most authors use only two to four moments. [76] shows that an approach by inertia moments combines a maximum of advantages.

$$\mu: \begin{array}{l}(E \to [0,1]) \to [0,1] \\ h \mapsto \sum_{\forall j} j.h(j)\end{array} \tag{2.2}$$

$$m_n: \begin{array}{l}(E \to [0,1]) \to [0,1] \\ h \mapsto \sum_{\forall j} (j - \mu(h))^n.h(j)\end{array} \tag{2.3}$$

The inertia moments interpretation depends on the chosen characteristic. For a greyscale histogram, average simply represents the average intensity; the standard deviation represents contrast, the third represents asymmetry and finally the fourth represents the flatness.

As for texture, first moments are also used, at least four. However, spatial orientation is important and other techniques must be used to determine granularity and directionality as well as other characteristics such as rugosity, periodicity, regularity, complexity, etc. [31]. Moreover, there does not seem to exist a better representation model for texture [57] [56].

However, histograms or inertia moments are still a too general representation of an image. In too large bases, several images may look very different but have very close histograms.

2.3.1.2 Colour models

The international standard for colour definition was established by C.I.E. in 1931. However, this colour model is not commonly used in the computer graphic industry.

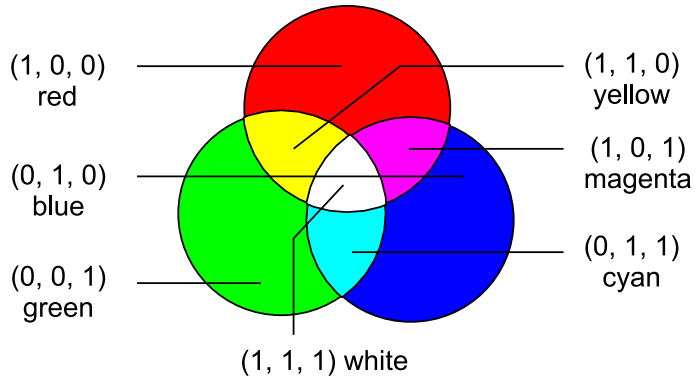

Figure 2.1 – The Red-Green-Blue colour model

For computer image manipulation, technical colour models like RGB or CMYK are preferred. Those models reflect the way pixels' colours are produced by the rendering device. Those technical models are not suitable for human intuitive colour representation. For instance, the *pink* colour is not easy to describe in term of red, green and blue combination. More accurate models, also said *perceptual* models are then used. The first of those models was proposed by A.H. Munsell in 1915.

The RGB colour model is also known as the additive model, since it is based on adding red, green and blue light. It is physically easy to create, this is why monitors and screens are using this model to produce a colour image. However, the red, green and blue components have no meaning for a human observer, and semantic information is poorly separated by this model. Consequently, this model is not adapted to information retrieval.

The HSV colour model is recognised to be one of the most perceptually evident for users [25]. HSV stands for Hue, Saturation and Value. All those components are immediately understandable as they reflect the way artists compose their colour: they first choose the Hue of the colour from different tubes, next they set the saturation by adding white and finally set the value by adding some black. In this model *pink* is seen as a red hue with some white in it to decrease its saturation. In the HSV space, this description is represented by the vector $\textit{pink} = \langle 0.0, 0.3, 1.0 \rangle$, with:

- $\textit{pink}.Hue = 0.0$: hue is defined, on the chromatic circle, as an angle in $[0, 2\pi]$ where 0 means *red*;
- $\textit{pink}.Saturation = 0.3$: the saturation scale ranges from 0 to 1;
- $\textit{pink}.Value = 1.0$: the value is defined on $[0, 1]$.

This model suffers one drawbacks when it comes to images retrieval: the hue property is not linearly perceptual. It means that zone colours such as red or blue represent a large part of the spectrum while yellow or green are represented by a very small portion. The consequence is that if the application does not take this into account, colours like red or blue are over-represented.

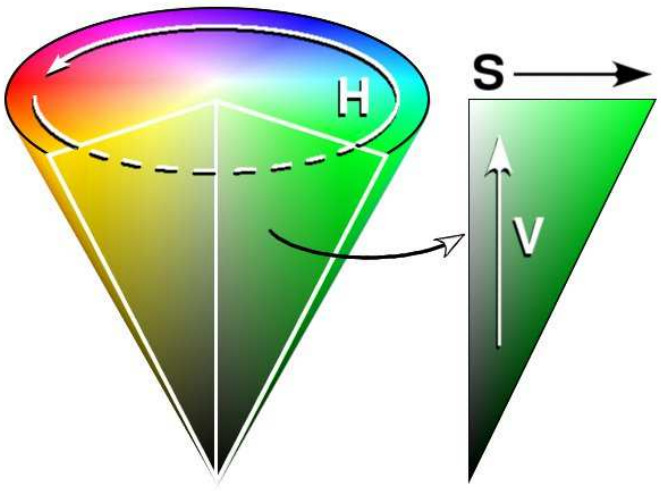

Figure 2.2 – The Hue-Saturation-Value colour model

Figure 2.2 shows a representation of the HSV colour model. It has been represented as a cone where the radius represents the hue; it also shows a slice of this cone for a hue corresponding to a green value.

2.3.1.3 Medium Level Data

Information extracted until now are mainly about the image in its whole. Still, the most relevant information after colour is the spatial disposition of main colours. This is a part of the pre-vision process, that is not conscious. Indeed, these colours generally correspond to real world objects. Without pretending this precision level, an image may be separated into interesting parts, with a minimum of hypothesis.

In order to provide more precise informations on spatial arrangement of the pixels of an image, we can first make minimal hypothesis on their composition. Then, we can consider that in the most of the cases the main object is close to the centre of the image [19]. Anyway, eye is attracted to the centre [28]. We can expect this property for images build in respect of the usual photograph rules by segmenting the images along the strength lines (segmenting into thirds, horizontally and vertically) [75]. To improve spatial informations relevance, a recursive segmentation of the image (quad-trees10 [65] [33]) may be performed until reaching a depth or homogeneity level fixed before-hand. Finally, more advanced techniques allow to express and exploit spatial concentration of colours: auto-correlograms [29], retro-projection [8], Ragon transformation [82], Delaunay triangulation [79], etc. To go further, one must use image analysis techniques. The main processes are segmentation and border extraction [3].

Segmentation techniques, very numerous, are used to extract regions of an image featuring a certain homogeneity, regarding a given criteria. Usual criteria are colour or greyscale homogeneity (cognitive studies also showed that the human eye is particularly sensible to large homogeneous colour zones [28] [5]) and texture homogeneity. Manual or semi-automated ver-

sions (flooding filling from a point chose by user, active border [19]) provide significant regions, possibly annotated with semantic informations, but at a high cost.

Every histogram process can be extended to regions. However, using regions one can take profit of new characteristics to improve the search results.

To each r region, one can associate additional characteristics related to shape and spatial arrangement:

- surface (s_r), perimeter (p_r) and orientation (θ_r);
- absolute position: barycentre ($C_x(r) = \mu(h(r))$ and $C_y(r) = \mu(h(r))$), minimal bordering rectangle (x_r, y_r, l_r, h_r considering or not θ_r);
- relative position: Euclidean distance to the centre of image ($C_c(r) = L_2((\frac{h}{2}, \frac{l}{2}), (C_x(r), C_y(r)))$);
- shape: stretching, eccentricity or elongation ($\frac{l_r}{h_r}$), rectangularity (such as $\frac{s_r}{l_r \times h_r}$), circularity or compacity ($\frac{4\pi s_r}{{p_r}^2}$, Fourier transform approximations [7], moments [31] or angles and tangent vectors [19]).

Moreover, binary (or even n-ary) characteristics may be associated between regions:

- comparisons on different properties;
- spatial relations: with (1) Allen relations [2] (*before, meets, overlaps, starts, during, finishes* and *equals*), (2) parametric coordinates $(\rho, theta)$ [81], or (3) relation graphs between objects of the same image, and even simple bi-dimensional adjacency histograms on their respective colour [24].

2.3.1.4 High-Level Data

High-level data are considered by some authors as being the most useful to perform relevant retrieval in an image database [9]. The principles of high-level data indexing consists in providing semantic descriptions of the scene and real world objects that can be found in this scene. Unfortunately, semantic usually has to be provided manually because it requires a comprehension of the scene, except in particular case described in previous part.

The keyword-based approach [8] is the most generic alternative to describe objectively as well as subjectively, intrinsically as well as extrinsically an image content:

$$C_M : I \rightarrow 2^{A^*} \tag{2.4}$$

Other models can hardly be compared. However, one may distinguish informations about (1) image regions and perceptual informations associated, (2) real world objects and semantic informations associated, and finally (3) establish a correlation between (1) and (2).

EMIR2 [43] introduces an oriented structured graph describing composite structure of objects (a house being a compound of a roof and walls...). *Spatial* informations are located on nodes (points, segments, polygons...) and arcs (metrics: close, far, vectorial: north, south, east, west; and topological: cross, overlap, disjoint, in, touch) of a description graph of the two-dimensional scene. Finally, a *symbolic* description separates formatted attributes (author, size...) from generic concepts. Note that this higher level proposition does not recognise objects in the image; only conceptual objects are described. Thus, there is no effective correlation.

It is also targeted to applications where semantic has an important role. $EMIR^2$ also includes relevance notions and uncertainness.

On the other hand, [44] uses a simple segmentation where each region is associated to a colour attribute. Regarding semantic description, an object-oriented approach is used (class, inheritance and aggregation). A link between two levels is established by a function associating a semantic object to one or several regions. Then, a query language is provided on this data structure.

CORE [84] goes further by providing several interpretations (concepts compounds) for the same property (measure compounds) of an image. An interesting application is the STAR system providing retrieval on company logos. These logos do not necessarily represent a real world object, but they do have a symbolic interpretation.

These few propositions are enough to show how approaches may differ in details, even if there are similarities at a high abstraction level.

In the case of IDB, we propose to use directly the application schema to obtain semantic informations, more generic. If $S =< C, A >$ is a very simplified database schema, where C is a set of classes and A a set of binary associations $A \subseteq C \times C$ between these classes, then one can define the C_S property associating to each instance the set of classes to which it is linked:

$$C_S : \begin{array}{l} I \rightarrow 2^C \\ i \mapsto \{c' \in C | i \in c \wedge i' \in c' \wedge (c, c') \in A \wedge (i, i') \in (c, c')\} \end{array} \tag{2.5}$$

One can, for instance, find the set of images that could represent urban landscapes by calculating $\{i \in I | City \in C_S(i)\}$.

2.3.2 Similarity measures

"To be intelligent is to find similarities" [14]. At one end, the first mathematical transformations are geometrical transformations, but they are too strict for our purpose. At the other end, topological similarities are too loose since a cup and a ring would be similar. To deal with similarities, one must define such a concept from gradual measures.

A similarity measure is usually define from a distance:

$$d : C \times C \rightarrow \mathbb{R} \tag{2.6}$$

respecting three axioms:

– auto-similarity: $\forall x \in C, d(x, x) = C$ (C being a constant);
– symmetry: $\forall (x, y) \in C^2, d(x, y) = d(y, x)$;
– triangular inequality: $\forall (x, y, z) \in C^3, d(x, y) + d(y, z) \geq d(x, z)$

Distances are numerous in the literature, defined by scalar values, set values, vectorial values, etc.

Unfortunately, several experiences showed that human perception is not comparable to a distance. Usually, it respects neither symmetry, nor even auto-similarity [66].

The general method should be as follows: whenever we want to express a perceptual similarity in a metric space, we have to define a it as a non-trivial, monotonic, and non decreasing function f, on an underlying appropriate distance:

$$s : \begin{array}{l} C \times C \rightarrow [0,1] \\ (c,c') \mapsto f(d(c,c')) \end{array} \tag{2.7}$$

2.3.2.1 Combination of measures

Another difficulty is how to combine several measures, often defined on different domains and sometimes with various weights. However, every experimental result shows that considering several characteristics at the same time gives better results than with a unique property [58] [49].

Considering several characteristics in the same time may be done at the time where characteristics are collected. [54] builds multi-dimensional histograms combining orthogonally several characteristics defined around a pixel (for example colour and intensity gradient). [73] operates a segmentation of images simultaneously in the colour domain and texture domain, in this way Savannah may be distinguished from lion's hair by using texture information.

In a general case, independently established measures will need to be combined. There are at least four difficulties:

- different values domains;
- there are qualitative data;
- data may not be independent from each others;
- the relative importance of properties is neither equal nor constant.

Literature presents several combinations methods, with their own merits and drawbacks. We classify them into four families:

- based on vote,
- measures in a vectorial space,
- probabilistic approach,
- and fuzzy logic.

Combination based on vote Several measures may be easily combined by voting techniques [70]. Its simplicity is attractive. Images are classified independently on different characteristics. Combination is done from the average rank, or median rank. Best and worst may be eliminated to gain robustness.

Using this technique, one can combine characteristics defined on different domains. Importance of characteristics may be weighted.

Aggregation in a vectorial space Techniques based on measures in a vectorial space are the most common ones. Characteristics have to be quantifiable. Then, each characteristic is an axis in a multi-dimensional space, and a vector of characteristics is a point in this space.

This approach can be applied as well to a multi-valuated criteria (such as histogram) that one wants to reduce as to several independent measures. In the general case, it becomes a tree of properties. It can even become recursive if the images are organised into compound objects.

When data can be assimilated to vectors, which is often the case for histograms, Minkowsky distances are often used:

$$L_p : \begin{array}{l} C^{\mathbb{N}} \times C^{\mathbb{N}} \to \mathbb{R} \\ (v, v') \mapsto (\sum_{\forall j} (v_j - {v'}_j)^n)^{\frac{1}{n}} \end{array} \tag{2.8}$$

Where $p \geq 1$. Most common distances are particular cases of Minkowsky distances: L_1 is the Manhattan distance, L_2 the Euclidean distance, and $L_\infty = max\{|x_i - y_i|\}$.

An important problem is *orthonormality* of the vectorial space. Considering for example a greyscale histogram, it is obvious that entry that are close from each others are almost equivalents. [76] shows histograms that are similar for human eye but can be made completely different by a simple translation.

The quadratic distance, supposedly found by Mikihiro Ioka from IBM laboratories in Tokyo, seems to solve that problem:

$$d_A^Q : \begin{array}{l} C^{\mathbb{N}} \times C^{\mathbb{N}} \to [0, 1] \\ (v, v') \mapsto \sum_{j=1}^{n} \sum_{k=1}^{n} a_{jk} d(v_j, {v'}_j).d(v_k, {v'}_k) \end{array} \tag{2.9}$$

It provides a correlation between axis as a symmetric matrix (in order to respect distance axioms), its diagonal terms being 1 (each entry being similar to itself). [27] and [73] use a matrix such as $a_{i,j} = (1 - \frac{d_{i,j}}{d_{max}})$ where d is the euclidean distance in the corresponding colour space and d_{max} the normalisation factor.

Note that L_2 is a particular case where the correlation matrix is identity. Another particular case is a binary matrix, equivalent to creating equivalence classes.

The only drawback of quadratic distance is the linear time complexity ($O(n)$), but dependencies between characteristics inside a given characteristic are common.

When independence of components is established, simpler measures fulfil. VisualSEEK [73] uses a simple sum between spatial distance (being itself a simple sum) and the colour distance (an adapted quadratic distance). [9] also proposes a sum of absolute distances between average and standard deviation, respectively on four forth and three channels of image.

Vectorial approach makes easy to take weight into account. As for histograms, it becomes especially useful when inertia moments are exploited. Indeed, average is more important than standard deviation, itself more important than next inertia moments. For example, two images with a different average colour are unlikely to be similar unless all other moments are close, which can be actually a translation of the colour spectrum.

A sum of quadratic differences, weighted by the inverse of corresponding variance, is used for texture (granularity, contrast and directionality) and the form in QBIC [16]. Note that it is possible to take into account weights that are internal to modalities in an histogram in order to compare only a subset of modalities (implicitly the case in simple interfaces where only a few dominant colours are specified).

Probabilistic aggregation Using probabilistic theory is an attractive idea [53] [12] [53] [49]. It allows to represent dissimilarities since dissimilarities are linked to the values appearance frequency, naturally presenting the shape of a function on a distance. [49] makes a trivial Gaussian hypothesis to reduce the distance on a characteristic C to a quasi-probability (statically normalised):

$$d'_c : \begin{array}{l} C \times C \to [0-\epsilon, 1+\epsilon] \\ (c, c') \mapsto \frac{d_c(c,c') - (\mu_c - 3\sigma_c)}{6\sigma_c} \end{array} \tag{2.10}$$

Implementation difficulties are known. To simplify, authors usually suppose that characteristics are independent to evaluate the similarity of a conjunction as a product [49] [9]:

$$s : \begin{array}{l} C^{\mathbb{N}} \times C^{\mathbb{N}} \to [0,1] \\ (v, v') \mapsto \prod_{i=1}^{n} s(v_i, v'_i) \end{array} \tag{2.11}$$

Of course, this is usually false, but according to experiments this is an acceptable approximation.

Unlike the vectorial model, weighting characteristics does not seem to be possible. Moreover, characteristics such as regions and their spatial relations cannot be taken into account neither in a vectorial point of view nor in a probabilistic point of view. Another measure is required.

Fuzzy aggregation Fuzzy aggregation [44] [53] fits more needs than probabilistic theory. It is even possible to combine quantitative data with qualitative data, if both are associated to fuzzy characteristic functions:

$$\mu_j : C \to [0,1] \tag{2.12}$$

[66] uses a proposition from Tversky approximating similarity between two binary stimuli sets as a weighting function f representing overlap and differences: $s(E,F) = \alpha f(E \cap F) - \beta f(E$
$F) - \gamma f(F$
$E)$ where E and F are strict sets. The f function is chosen as the cardinality of the fuzzy set:

$$f : \begin{array}{l} C^{\mathbb{N}} \to [0,1] \\ v \mapsto \sum_{i=0}^{n} \mu(v_i) \end{array} \tag{2.13}$$

Intersection and difference operators are naturally extended from fuzzy logic's min and max operators. Of course, this solution is just a candidate among what fuzzy logic proposes. Note that weights are managed by modificators. Also note that dependencies between characteristics have been studied [84] [66].

2.3.3 Feedback loop

In the retrieval process, the results' quality measure is not objective, it strongly depends on the user. Using feedback loop, one can evaluate the intermediate results' quality before to output the actual results. As for any "smart" human-computer interaction, the goal is to discover what characteristics matter to the user.

Techniques used in textual retrieval, based on the vectorial model [64], build a suite that have been proved to converge in a finite time but with no guaranteed limit to a dichotomy between relevant data and the rest [20]:

$$v_{n+1} = v_n + \alpha \frac{\sum_{v \in P} v}{|P|} - \beta \frac{\sum_{v \in N} v}{|N|} \tag{2.14}$$

Where α and β are weights to determine, P and N respectively the subset of relevant element and the subset of non-relevant elements.

A variation from [32] is based on a principal components analysis to keep significant characteristics associated to images presented as examples and reject significant characteristics associated to counter-examples.

To speed-up convergence, learning techniques should be used. In the probabilistic model, [49] supposes that values that can take different characteristics into account respect a Gaussian distribution, and that these characteristics are independent to propose:

$$P_{\mu,\sigma}(v|i \in pertinents) = \prod_{j=1}^{n} P_{\mu_j,\sigma_j}(v|i \in pertinents) \tag{2.15}$$

The problem is equivalent to finding the best values for average and standard deviation. For this, a gradient descent optimisation algorithm is proposed; it constructs a hypothesis suite. These are linked by a factor $\lambda \in]0,1[$ (rather close to 1) that progressively reduces variance. Average is adjusted regarding relevant images, while a minimal number of relevant images is included under the defined Gaussian curve.

A more generic approach is proposed in [41]. It is based on a logical formulation of a query taking into account regions of an image (own characteristics and links between them). Determining a query that keeps examples and rejects counter-examples is proved to be an NP-complete problem (reduction to the minimum cover problem [22]). A genetic algorithm is used to solve it.

CHAPTER 3

NAVIGATION THROUGH AN IMAGE COLLECTION

Due to the visual nature of the *image* media type, navigation appeared as a powerful and user-friendly way to find images in a collection or a database. Not the image media is a visual one, thus requiring feedback from the user to determine relevance of results to his or her needs, but being a still media (unlike video), to represent an image using a small space is quite easy.

The nature of image and the popularity of the world wide web resulted in a lot of proposals using navigation to find images from a large collection or a database. While most proposals will fit in the first one, we divide these proposals into two categories:

1. Proposals using navigation only for user-interface, but keeping a classical retrieval method (queries or relevance-feedback) for the engine, and
2. proposals using a navigation structure as a search concept, *i.e.* using the same structure for indexing and interaction with the user.

3.1 Disposing Images in a Space

A lot of proposals dispose images in a space, usually using distances (thus similarity search). Display device (screen) being usually two-dimensional, most proposals [69] use a two-dimensional space: a plan.

If images are to be displayed on a space, one may intuitively want to make use of the *location* information when it is available. This approach is used by Geobloggers (http://www.geobloggers.com, illustrated in figure 3.1. Geoblogger is basically a world map where images appear at the place they have been taken. Similar proposal also use the *time* information. Combining space and time is usually interesting since they usually represent *events*. Indeed, pictures taken the 6 of August, 1945 in Hiroshima will obviously represent pictures of the atomic bomb that hit this city. Such images will have a strong semantic link between each others. Photographs taken at either a different place or a different time may have a completely different meaning.

Kaester *et al.*'s work [34] proposes to combine several input methods to search for images, including touch screen (to select parts of images or perform gestures) and speech recognition.

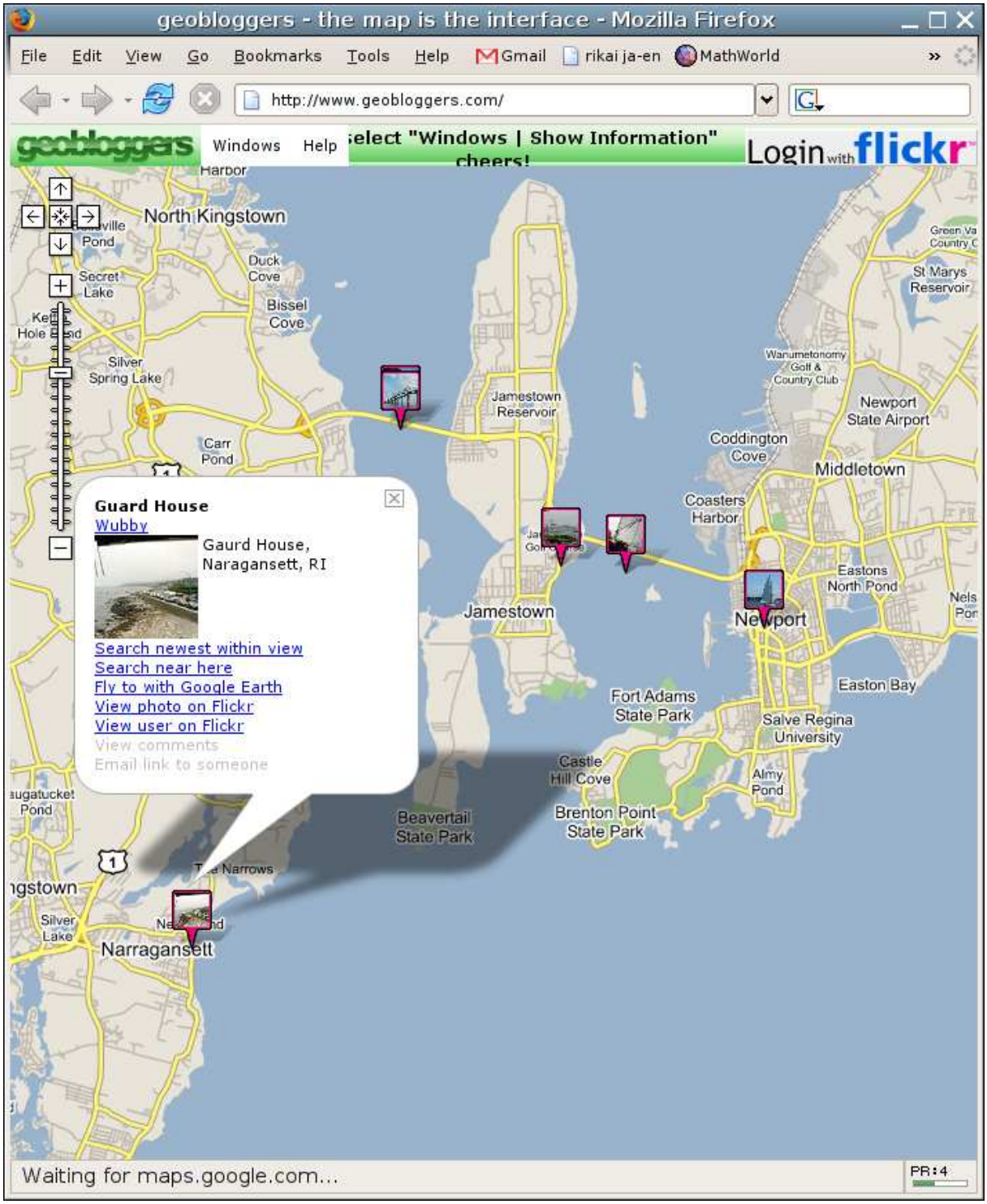

Figure 3.1 – Images Displayed According their Geographic Location

Figure 3.2 – A Modigliani's Painting with Similar Images [69]

By using these non-classical input methods Kaester could produce a graphical interface that makes the user feels like he or she is navigating the image database. This system is however still based on similarity search: the user will actually select images or parts of the images for the system to find images similar to these samples. The collection is stored by using multi-dimensional indexing techniques.

3.1.1 The El Niño Project

In their project El Niño, Santini *et al.* worked on integrating browsing and querying [69]. Their proposal is a set of search engines connected by a mediator that dispatches the queries to the search engines, collects the results and displays them to the user. Images are arranged on a two-dimensional plane, and the user interacts with the system mainly by two ways:

– By clicking on an image, the user asks the system to move this image to the centre. From the user's point of view, he or she is moving inside the collection; from the system's point of view, the user is launching the query "show the images similar this one". This is illustrated in the figure 3.2 where images are organised around the Modigliani's painting (in the middle).

– By drag-and-dropping images, the user teaches the system similarities that were not present. The user can then tell the system that from his or her point of view, two images are similar. In other words, this is a user-personalisation process.

3.2 Using a Navigation Structure

A different approach is to navigate on a structure that is built before-hand. This is basically what is used inside structured documents, or human-edited directories such as *Yahoo!* (http://www.yahoo.com) or *dmoz Open Directory Project* (http://dmoz.org).

These structures can be of different shapes, and according to this shape different tours [4] may be performed on it.

– The simplest model is the *linked lists* model: the user can navigate using links to the *next* and *previous* element.
– A more elaborated one is the *hierarchical* model: a second level, the parent/child relationship, is added. The user can go *specific* by choosing a child node or go *general* by selecting a parent node. For example, in the *dmoz Open Directory Project*, let's consider a user browsing the category `Top: Computers: Software: Databases`. He or she can for example go up to the `Top: Computers: Software` category or choose to go down in one of the subcategories, for example `Top: Computers: Software: Databases: Object-Oriented`. This kind of hierarchy can be assimilated to an anthology.

In these structures, a given node usually have a single parent node. Multiple parents is simulated by creating links between categories. For example, `Top: Computers: Programming: Languages: Database` appears as a link in the category `Top: Computers: Software: Databases`.

Finally, a graph allows to define a more complex structure where the user is not limited to go specific or go back from where he or she is coming from. However, one should be very careful while using a graph structure: if it is too complicated, the user can literally get lost in the structure and thus be unable to find what he or she is looking for.

3.2.1 Navigation Structures for Multimedia Data

Most of the individuals or companies with limited needs organise their images using directory and files. Usually, they use one directory to store their collection and create a new directory for each event. By doing that, they are creating a structure to organise their data.

Tools like Google's Picasa (http://www.picasa.com) extend this model by proposing to add tags or titles to images, and propose navigation and retrieval more adapted to images that what a general purpose file browser provides. This is useful for users who need only a rough classification of images; however most users will not care about adding tags or titles individually to images. This will be limited to a per-directory tagging.

In order to organise images while asking the user for a minimum of interaction, the use of content-based information is required. However, while containing latent semantic, content-

based information usually have no direct meaning for a user; moreover high-dimensionality is usually involved. A navigation structure based on content-based should thus be build very differently from a navigation structure based on an anthology.

Navigation through an image collection is usually done either using a similarity search as a base, or on a structure created by a human operator. There have been almost no research work on a structure automatically created from content-based information, but this will be the goal of the current proposal.

PART II

Efficient Structures for Navigating an Image Collection

CHAPTER 4

NAVIGATING AN IMAGE COLLECTION USING GALOIS' LATTICES

This chapter presents our first proposal, Galois' lattices used directly as an indexing and navigation structure. First, we will describe the meta-model we use in the section 4.1. This model will be used not only in this first proposal but also in the extensions that will be presented in the next chapters.

Then, in the section 4.2, we will present our proposal based on *concept lattices*, a useful graph structure, helping the user to browse an image collection organised before-hand.

Finally, the implementation of this work will be detailed in section 4.3 and the results of the experiments we conducted will be presented in the section 4.4.

4.1 A meta-model for navigation-based "retrieval" on images

Considering the MPEG-7 model presented in section 2.3.1, we basically take into account the levels 1 to 4 in order to focus on content-based retrieval. As for colour information, we chose the HSV (Hue-Saturation-Value) colour model from the models presented in section 2.3.1.2. The HSV colour models combines a good perceptual fidelity to a low processing cost; the problem of the non-linearity of hue will introduce no bias since the definition of fuzzy subsets on this axis will take it into account.

Also, we restrict our attention to "standard" images, i.e., two-dimensional, rectangular, without transparency channel, that are not bounded to any particular area (e.g., satellite or medical images). Specific kinds of images require adequate descriptions that can be far different from the ones introduced hereafter.

A lot of properties can be used to represent an image, as detailed in the section 2.3.1. However, considering too much properties at once generally suffers some drawbacks. Firstly, query performances degrade rapidly, the so-called "high dimensionality curse problem." This is not a problem here since building the hypertext of images is done off-line. Hence, navigation offers optimal performances both from the processing and storage point of view, because the links are "hard-coded." Secondly, common weighted queries are very sensitive to correlated features

[72]. The technique that we use, namely concept lattices (detailed later in the section 4.2), is insensitive to this problem due to the absence of weights, and even of distances.

The sequel of this section presents some reasonable choices to provide a fuzzy linguistic description of an image through its colour features and general properties. There are other properties such as texture (briefly cited in section 2.3.1.1) but we purposely renounce to use them for cost and performances reasons.

Definition 1 (Description Space)**.** *The description space $\mathcal{D}$ consists of the union of several sub-descriptions:*

$$\mathcal{D} = \mathcal{D}_{area} \cup \mathcal{D}_{orientation} \cup \mathcal{D}_{elongation} \cup (region \times \mathcal{D}_{hue}) \cup (region \times \mathcal{D}_{saturation}) \cup (region \times \mathcal{D}_{intensity}) \quad (4.1)$$

4.1.1 Fuzzy linguistic labels for colour

Fuzzy linguistic Labels (FL) like *pink* need to be described over the colour domain, namely, the HSV space. Each FL c is associated to a membership function μ_c with values within $[0, 1]$, reflecting how well the FL c describe the colour of a particular image or region of an image. For instance, the *pink* label could be represented over the HSV colour space thanks to a membership function $\mu_{pink}(h, s, v)$ with high membership values associated to colours with a hue close to red, a low saturation and a very high value.

Definition 2 (Fuzzy Subset)**.** *A fuzzy subset A of a set X is defined by a membership function (or characteristic function) μ_A where $\forall x \in X, \mu_A(x) \in [0, 1]$. $\mu_A(x)$ represents the membership degree of x in the fuzzy subset A.*

This general approach [63] allows to represent any FL with a fuzzy set directly defined over the colour domain. Coverage of the colour space is required to ensure that any image will have at least one representation in terms of linguistic descriptors. Allowing FL to be defined separately from each other induces a greater flexibility, but the domain coverage has to be checked after the user is done with setting all label definitions. It is likely that some holes will remain uncompleted.

To overcome this, another method has been used. With this method, any colour linguistic label c is formed from the concatenation of elementary labels $c.h$, $c.s$ and $c.v$ defined on each of the H, S and V dimension. The membership function associated with c is computed as a conjunction of the elementary label membership functions:

$$\mu_c(h, s, v) = \mu_{c.h}(h) \otimes \mu_{c.s}(s) \otimes \mu_{c.v}(v)$$

where $\otimes$ represents a T-norm (max for instance) used to compute the conjunction of the individual channels. In this test, we used the following partition:

- Hue : *red, orange, yellow, green, cyan, blue, magenta*
- Saturation : *vivid* and *dull*
- Value : *dark* and *bright*

Colour labels are formed by combining those terms (e.g. *vivid bright red*). Under a certain value level, colour is perceived as black. In the same way, colour is perceived as grey or white, under a certain saturation level. Hence, to the above collection of colour terms, we add *black*, *dark grey*, *bright grey*, and *white* resulting in a total of 32 individual terms. This number is in the range of the QBIC system which defines a set of 25 colours.

This method is much simpler from the user's point of view, and the coverage of the HSV space is guaranteed as long as each dimension is well covered, which is trivial to achieve.

4.1.1.1 Zone colour characterisation

Colour perception results from the juxtaposition of individual pixels. The perceived colour of an arrangement of pixels ranges from uniform pure colour to complex colour arrangement without dominating colour.

Considering our linguistic representation of colours, each pixel colour is expressed in terms of colour labels with different weights. For a pixel, the weight is the membership degree of its colour to the fuzzy set associated to a colour label. For instance, in our paradigm of representation, a pink pixel could be defined as two colour labels:

– *vivid bright red* with a membership degree of 0.1;
– *dull bright red* with a membership degree of 0.9.

Considering a region S as a collection of adjacent pixels, the relative importance $\tau_S(d)$ of a colour label d inside S, is computed as the sum of membership degrees $\mu_d(p)$ of each pixel:

$$\tau_S(d) = \frac{\sum_{p \in S} \mu_d(p)}{\sum_{d' \in \mathcal{D}} \sum_{p \in S} \mu_{d'}(p)} \quad , \tag{4.2}$$

with $\mathcal{D}$ the set of all colour labels.

4.1.2 Syntactical division

An image usually contains several real-world objects, and separating these semantically independent objects leads to a more accurate description of image colours, thus considerably improving the results' quality. Segmentation algorithms perform this separation by identifying homogeneous zone, using colour informations and sometimes texture informations also.

However, due to:

1. the high algorithmic cost of the segmentation algorithms, and
2. the failure of these algorithms to recognise real-world objects made of several parts of different colour and texture,

we relied on a syntactical division of the images rather than a real segmentation.

Considering general photographic pictures, the main subject often stands in the centre and the surrounding areas represent the image background. In addition, colour homogeneity is expected to be enhanced if smaller zones are considered. In a landscape picture, for instance, the

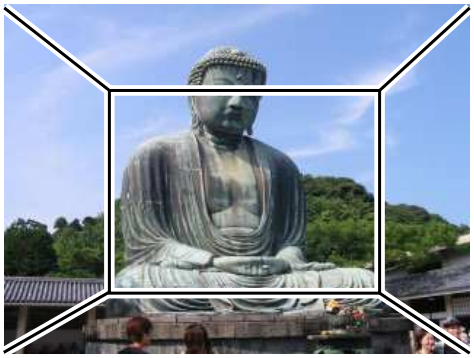

Figure 4.1 – Syntactical Division: Big Buddha

sky is likely to have blue or grey hues, while the ground will probably be green. Among different divisions we tried, a division into five trapezoids seemed a good model to separate semantics in the general case.

These five trapezoids are respectively the centre, the left, the right, the top, and the bottom part of the image. Figure 4.1 shows an sample image featuring Kamakura's Big Buddha divided into trapezoids. The central zone covers 49% of the total surface and the four surrounding zones are trapezoids, the wideness of which is 15% of the image wideness.

4.1.3 General geometrical measures

Along with colour, we consider geometrical measures to represent the *area*, the *orientation*, and the *elongation* of the image.

The use of orientation and elongation for image search is quite obvious: an image wider than high is usually called a "landscape" image precisely because most landscape images have these proportions. Even when it is not an actual landscape, a landscape-oriented image usually represents several real-world objects, a very large image being usually a panoramic view. On the contrary, an image higher than wide is called a "portrait" because portraits usually have these proportions. When it is not a portrait, a portrait-like image usually represents a close-up of a unique real-world object. Thus, there is an implicit correlation between the orientation and the elongation, and parts of the semantics of the image.

The area is also an important measure since users may be interested by retrieving images of a certain size according to their needs.

4.1.3.1 Numerical Features

First, $\mathcal{D}_{area}$, $\mathcal{D}_{orientation}$, and $\mathcal{D}_{elongation}$ are format informations (MPEG-7 level 1), related to the whole image i, based on the following *scalar* functions:

- $area(i) = width(i) \times height(i)$,
- $orientation(i) = \frac{1-\cos(2\alpha(i))}{2}$, and

– $elongation(i) = |\frac{4\alpha(i)}{\pi} - 1|$.

where $\alpha(i) = \text{atan}\frac{width(i)}{height(i)}$ is the angle of the diagonal.

These formulæ have been chosen among various alternatives. The orientation of the image (i.e., portrait or landscape) is independent of the area of the image.

Also, using an absolute value removes the correlation between orientation and elongation measures: the covariance is null on $[0, \pi/2]$. (However, correlation is perfect per halves, i.e., on $[0, \pi/4]$ and $[\pi/4, \pi/2]$ independently, due to the functional dependency between orientation and elongation.)

4.1.3.2 Linguistic Variables

The introduced numerical features are used in turn to provide metadata on images as subsets of the following descriptions:

$$\mathcal{D}_{area} = \{tiny, small, medium, large, huge\} \tag{4.3}$$

$$\mathcal{D}_{orientation} = \{portrait, square, landscape\} \tag{4.4}$$

$$\mathcal{D}_{elongation} = \{none, standard, panoramic, elongated\} \tag{4.5}$$

$$\mathcal{D}_{hue} = \{red, orange, \ldots, cyan, blue, magenta\} \tag{4.6}$$

$$\mathcal{D}_{saturation} = \{vivid, light, pale\} \tag{4.7}$$

$$\mathcal{D}_{intensity} = \{black, dark, light, white\} \tag{4.8}$$

These discrete subsets are obtained from the corresponding fuzzy linguistic variables and fuzzy subsets, and subsequent thresholding.

Figure 4.2 illustrates the classical way to define the fuzzy subsets of a linguistic variable as trapezes or trapeze-like shapes, i.e., to each member of a sub-description $\mathcal{D}_i$ we associate an arbitrary membership function.

Definition 3 (Scalar Fuzzy Value)**.** *The fuzzy value of a scalar property with respect to a fuzzy subset A is directly given by its membership function:*

$$FV_i : \begin{array}{l} \mathbb{R} \times \mathcal{D}_i \to [0,1] \\ (x, A) \mapsto \mu_A(x) \end{array} \tag{4.9}$$

where $i \in \{area, orientation, elongation\}$.

Definition 4 (Vectorial Fuzzy Value)**.** *The fuzzy value of an histogram h with respect to a subset A is computed as follows:*

$$FV_j : \begin{array}{l} ([0,1] \to [0,1]) \times \mathcal{D}_j \to [0,1] \\ (h, A) \mapsto \int_0^1 h(x) \times \mu_A(x)\, dx \end{array} \tag{4.10}$$

where $j \in \{hue, saturation, intensity\}$.

Definition 5 (Fuzzy Description)**.** *A fuzzy description is a set of fuzzy subset names and corresponding non null fuzzy values:*

$$FD_i : \begin{array}{l} \mathbb{R} \cup ([0,1] \to \mathbb{N}) \to 2^{\mathcal{D}_i \times [0,1]} \\ p \mapsto \{(A, FV_i(p, A)) | A \in \mathcal{D}_i \wedge FV_i(p, A) > 0\} \end{array} \tag{4.11}$$

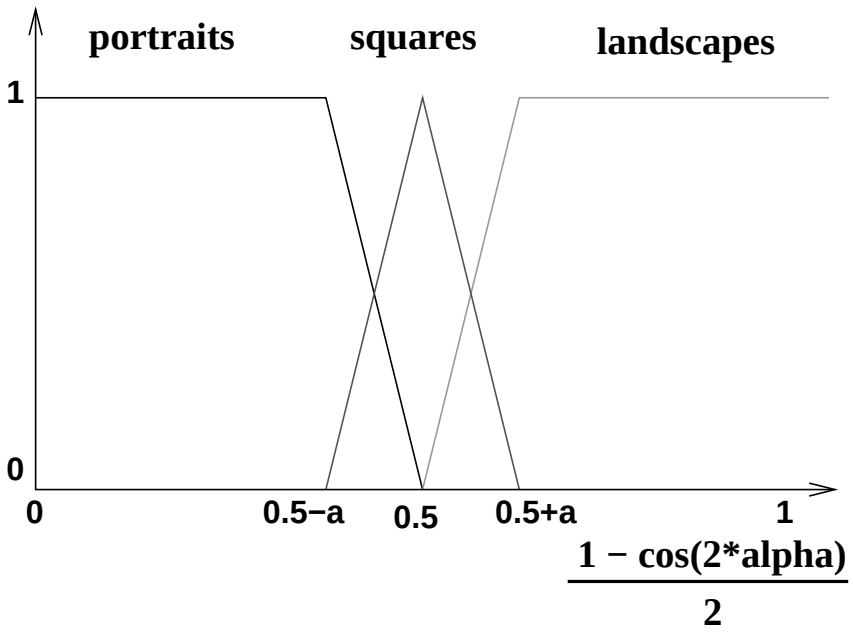

Figure 4.2 – The $orientation$ linguistic variable with its three fuzzy subsets

4.1.4 Binary model

In order to use a Galois' lattice, we have to remove the fuzzy membership degrees and produce a binary relationship. In this section we present a trivial way to remove the fuzzy membership degree, but in the chapter 7 we will propose a better way to build a Galois' lattice from a fuzzy relationship.

Definition 6 (Discrete Description)**.** *A discrete description is obtained from a fuzzy description and thresholds,* α_i*:*

$$DD_i : \begin{array}{l} \mathbb{R} \cup ([0,1] \to \mathbb{N}) \to 2^{\mathcal{D}_i} \\ p \mapsto \{A | (A,f) \in FD_i(p) \wedge f \geq \alpha_i\} \end{array} \tag{4.12}$$

For instance, the $elongation$ of an image could be *{ (standard, 0.3), (panoramic, 0.7) }* and would lead to *{ standard, panoramic }* if the α-cut is set to 0.3. Similarly, from *{ (red, 0.5),- (orange, 0.1),(blue, 0.4),(magenta, 0.2) },* we obtain *{ red, blue, magenta }* with a threshold of 0.2.

Definition 7 (Discrete metadata)**.** *Finally, the complete metadata description of an image is given by:*

$$M : \begin{array}{l} \mathcal{I} \to \in^{\mathcal{D}} \\ i \mapsto \bigcup_{d \in general} DD_d(d(i)) \cup \\ \quad \bigcup_{r \in region} \bigcup_{d \in cComp} \{(r, DD_d(h_r(r(i))))\} \end{array} \tag{4.13}$$

where:

- $general = \{area, orientation, elongation\}$
- $region = \{top, bottom, left, right, center\}$

– $cComp = \{hue, saturation, intensity\}$
– $r(i)$ *is the function that extracts the pixels of the part* r *of the image* i.

Exemple 4.1. $\mathcal{D} = \{medium, \ldots, standard\} \cup Colourset$
where $Colourset$ *is:*

$$\begin{pmatrix} red \\ \vdots \\ magenta \end{pmatrix} \otimes \begin{pmatrix} pale \\ \vdots \\ vivid \end{pmatrix} \otimes \begin{pmatrix} black \\ \vdots \\ white \end{pmatrix} \otimes \begin{pmatrix} top \\ \vdots \\ bottom \end{pmatrix}$$

Size of the Description Space The $area$, $orientation$, and $elongation$ linguistic variables can generate discrete descriptions of 0 up to 2 items, because each trapeze overlaps only its direct neighbour(s). However, a finer discrimination could lead to a higher degree of overlapping, denoted k. In addition, the expected level of overlapping depends on the corresponding threshold: in general, the higher the threshold is, the less overlapping can occur. Formally, the number of possibilities is bounded by:

$$P(\mathcal{D}_i) = 1 + A^{k_i}_{|\mathcal{D}_i|} \tag{4.14}$$

where 1 stands for the empty set and A^m_n (the number of arrangements of m items within a set of n objects) gives the number of cases of *consecutive* fuzzy values that are over the chosen threshold.

In contrast, the hue, $saturation$, and $intensity$ variables generate an exponential number of cases. For instance, the set of different hues that appear in an image can be any subset of all the possible hues, for a sufficiently low α-cut. Therefore, the number of possibilities is bounded by:

$$P(\mathcal{D}_j) = 2^{|\mathcal{D}_j|} \tag{4.15}$$

Finally, the size of the concept space associated to $\mathcal{D}$ is bounded by the following formula:

$$\prod_{i \in \{area, orientation, elongation\}} P(\mathcal{D}_i) \times |region| \times \prod_{j \in \{hue, saturation, intensity\}} P(\mathcal{D}_j) \tag{4.16}$$

With the current values, the size of the concept space is already $(1+5+4) \times (1+3+2) \times (1+5+4) \times 5 \times (2^7 \times 2^3 \times 2^4) = 10 \times 6 \times 10 \times 5 \times (128 \times 8 \times 16) = 49,152,000$. This is far beyond the size of the biggest image database that we know of, and gives a hint about the discriminative power of this technique.

However, we do expect several images to be grouped under the same description, even for small databases in order to achieve an actual classification. (The rare risk is to have a set a images that pairwise share only one property. This would produce a combinatorial explosion, i.e., a lattice with an exponential number of nodes.)

That is close to what we did observe on our example (see the section 4.4), the only problem is that the number of properties for an image is not equitably divided. Most of the images have an average number of properties.

4.1.5 Resulting models

In this section we described a fuzzy model for image retrieval, including colour informations as well as general shape informations (orientation, elongation, shape). For colour informations, a syntactical division of the image into five parts (top, bottom, left, right and centre) have been used in order to separate real-world objects in an efficient way.

We also derived a binary model (similar to a model based on keywords) by applying a threshold to the fuzzy model. However, the only reason to build a fuzzy model was not to derive a binary model from it. It will be used later:

- In the chapter 5, when will be described an additional clustering to improve the scalability. The fuzzy model will be used to perform the clustering process.
- In the chapter 7, techniques more elaborated than simple thresholding will be presented.

4.2 Navigation on a concept lattice

The images representation being defined, in this section we shall describe how to organise the collection so the user can browse it.

Browsing techniques are numerous, from mere guided and indexed tours [23, 30] to advanced tours [42]. Whatever technique is used, it requires prior structuring of the data. For instance, a guided tour corresponds roughly to a linked list and may have been constructed from a set by possibly selecting some items and specifying a given order between them.

Images are described by various properties, their so-called *metadata* detailed in the previous section. If we envisage only simple tours on simple properties, like the list of all the grey-level images, the user will be unsatisfied with lengthy lists. Combining several properties is unavoidable. We propose here for $Click^{Im}_{AGE}$ a *concept lattice tour* on the metadata of the images. In short, this structure allows, through mere clicks, either to focus on more and more constrained images (e.g., mainly blue, panoramic images with a strong texture), or on the contrary to have a fast access to general classes of images (e.g., just grey-level images).

First, in the section 4.2.1 we will present Galois' (concept) lattices and how to build them. Then, in section 4.2.3 we will detail our proposal of hypermedia browsing of such a structure.

4.2.1 Galois' Lattices

Definition 8 (metadata)**.** *A simple* metadata *structure is defined in the form of a binary relation:*

$$R : \mathcal{I} \times \mathcal{D} \tag{4.17}$$

where $\mathcal{I} = \mathbb{N} \times \mathbb{N} \rightarrow [0,1]^3$ *is the set of images, and* $\mathcal{D}$ *is a set of admissible descriptions.*

Note that admissible metadata vary from application to application. They can be related to the intrinsic content of the images, e.g., colour, or they can add some semantics to them, e.g., through mere keywords. This first definition remains voluntarily vague to permit several techniques to be used.

However, in the current state of the system, we restrict $\mathcal{D}$ to discrete values, using the model based on the colour information and the general shape presented en the previous section. A better usage of fuzzy descriptions is detailed in the section 7. We base our proposal on Galois' lattices. An image lattice is just another instantiation of a Galois' (or concept) lattice. (This mathematical structure has been largely exploited in the field of knowledge discovery [26], and of object-oriented hierarchy design.) Basically, it allows to create a special kind of directed acyclic graph, the nodes of which group a set of instances, i.e., an *extension*, and a set of descriptions, i.e., an *intention*.

We derive a *Galois' lattice* from this relation. The reader interested by a more detailed Galois' lattice formal description may read [26].

A lattice is an oriented graph, without any loop, and including an *inferior node* (no edge ends to this node) and a *superior node* (no edge starts at this node). In a *Galois' lattice*, nodes are pairs (X, X') where $X \subset \mathcal{I}$ and $X' \subset \mathcal{D}$. We note $\mathcal{C} = \mathcal{I} \times \mathcal{D}$, the set of pairs (possible nodes). These pairs must be *complete pairs*, defined as follows [26]:

Definition 9 (Complete Pair)**.** *A pair (X, X') is complete with respect to R if and only if the two following properties are satisfied:*

- $X' = \{d \in \mathcal{D} | \forall i \in X, (i, d) \in R\}$
- $X = \{i \in \mathcal{I} | \forall d \in X', (i, d) \in R\}$

Only maximally extended pairs (for which there is no other pair (X_1, X_1') such as $X \in X_1$ and $X' \in X_1'$) are kept.

Basically, that means that an image $i \in X$ features *at least* each property $d \in X'$ and a property $d \in X'$ is respected by *at least* each image $i \in X$. The X set is called *intention* and the X' set *extension*.

The following property may easily be demonstrated:

Property 1. *Given two complete pairs $((X_1, X_1'), (X_2, X_2')) \in \mathcal{C}^2$:*

$$X_1 \subset X_2 \Longleftrightarrow X_2' \subset X_1' \tag{4.18}$$

We can now define a *partial order* between pairs:

$$\forall (C_1 = (X_1, X_1'), C_2 = (X_2, X_2')) \in \mathcal{C}^2, C_1 < C_2 \Longleftrightarrow X_1 \subset X_2 \Longleftrightarrow X_2' \subset X_1'$$

This partial order is used to generate the lattice graph as follows: there is an edge (C_1, C_2) if $C_1 < C_2$ and there is no other element C_3 in the lattice such as $C_1 < C_3 < C_2$.

Exemple 4.2.
From $R = \{(img_1, blackbottom),$
$(img_1, yellowcentre), (img_2, blackbottom),$
$(img_3, yellowcentre), (img3, redtop),$
$(img4, yellowcentre)\}$

We derive: $r(img_2) = \{blackbottom\}$,
$r(img_1) = \{blackbottom, yellowcentre\}$,

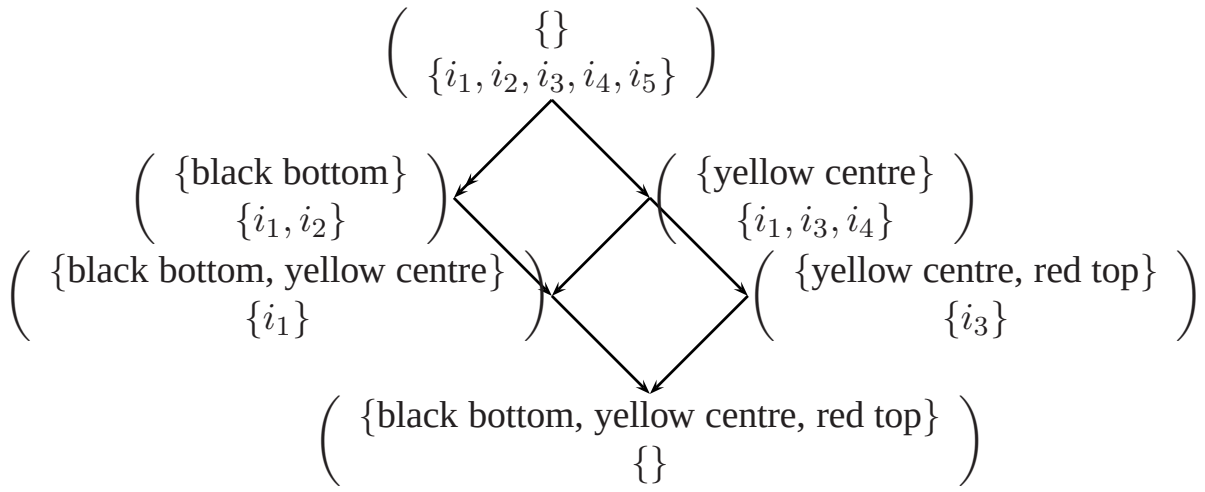

Figure 4.3 – An example of an image lattice

$r(img_3) = \{yellowcentre, redtop\}$ *and*
$r(img_4) = \{yellowcentre\}$*;*

and conversely $r'(redtop) = \{img_3\}$,
$r'(blackbottom) = \{img_1, img_2\}$ *and*
$r'(yellowcentre) = \{img_1, img_3, img_4\}$.

In addition, in a Galois' lattice, all the possible intersections for which at least a descendant node is non empty are added.

Intuitively, we are interested in the set of images that share *exactly* the same description, and moreover *at least* the same description.

4.2.2 Building a Galois' lattice

Since a Galois' lattice depends on global properties of the considered binary relationship, building a Galois' lattice is not a trivial problem. However, there have been a lot of research on this problem.

Ganter's *NextClosure* algorithm [21] is often cited as a reference algorithm, to which most authors compare their own proposal.

Later, Godin et. al. [26] proposed an incremental algorithm to build a Galois' lattice, with an empiric square time complexity ($O(n^2)$). Recently, Levy et. al.'s [37] proposed a parallel algorithm called ELL. This algorithm is interesting to build a very large lattice using a cluster of machines. However, this algorithm is not incremental, thus not adapted to our case where it should be possible to add images to an existing structure.

For this work, we chose Godin's algorithm [26] that is:

– Incremental, thus allowing us to update the structure,

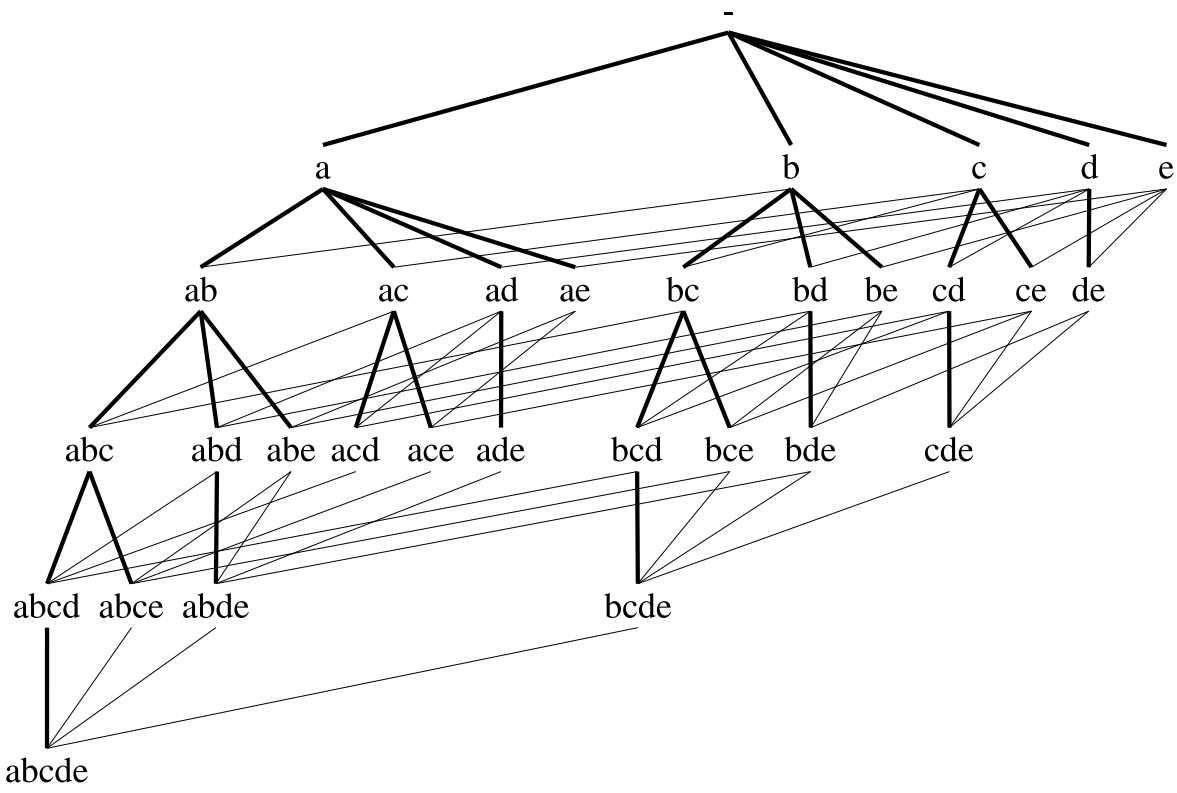

Figure 4.4 – A sample Galois' lattice as a (hyper-)cube of dimension 5 – including its prefix-tree (strong lines) –

– of square complexity ($O(n^2)$), which is good regarding the difficulty of the problem and the other proposals.

4.2.3 Using a Galois Lattice as a Navigation Structure

Our approach does not use Galois' lattices only for indexing but also for browsing itself, from the user's point of view. A set of XHTML pages is constructed from the Galois' lattice, that is stored either in central memory or in a database management system (DBMS). These pages being generated only after a new image is inserted into the structure, the user will browse a set of static pages; consequently, the navigation process is optimal ($O(1)$) if we neglect the time to load and display images.

For one node of the Galois lattice, corresponding to an *intention* (a set of descriptions) and an *extension* (a set of images featuring at least these descriptions), one XHTML page is generated.

The intention is not displayed to the user. We consider that the user should make up his or her choice by visualising sample images, not by abstracting his or her needs. Consequently, a node is only represented by a set of images.

Rather than using the complete extension of a node to represent it, we use what we call the *reduced content* of this node.

Definition 10 (Reduced content)**.**
The reduced content Y of a node (X, X') is defined as:

$$Y = \{i \in \mathcal{I} | \forall d \in X', (i, d) \in R)\}$$

It means that the reduced content of a node is the set of images that features all the descriptions of its intention, and only these descriptions. Note that since the extension X is the set of images featuring *at least* the descriptions in the intention X', Y is subset of X. The calculation of the reduced content has been integrated into Godin's incremental lattice building algorithm [26], and does not change its complexity.

A node is represented as follows:

– When the reduced content of a node is not the empty set, these images are used to represent this node.
– When the reduced content is the empty set, the node's representation is built recursively from the reduced content of its children nodes.
– In the top of the page, links to father nodes using these nodes' representations are provided.
– Similarly, in the bottom of the page, links to children nodes using these nodes' representations are provided.

4.3 Implementation

From a performances point of view, constructing a concept lattice is not an easy task. Basically, the complexity is in $O(n^2)$ where n is the number of nodes, and theoretical improvements

$lattice \leftarrow \emptyset$
for $image \in \mathcal{I}$ do
--$center \leftarrow center(i)$; $top \leftarrow top(i)$; $bottom \leftarrow bottom(i)$; $left \leftarrow left(i)$; $right \leftarrow right(i)$
--$metadata \leftarrow \emptyset$
--$metadata$.Insert($DD_{area}(area(image))$)
--$metadata$.Insert($DD_{orientation}(orientation(image))$)
--$metadata$.Insert($DD_{elongation}(elongation(image))$)
--$metadata$.Insert("center " + $DD_{intensity}(h_{intensity}(center))$)
--$metadata$.Insert("center " + $DD_{saturation}(h_{saturation}(center))$)
--$metadata$.Insert("center " + $DD_{hue}(h_{hue}(center))$)
--$metadata$.Insert("top " + $DD_{intensity}(h_{intensity}(top))$)
...
--$metadata$.Insert("right " + $DD_{hue}(h_{hue}(right))$)
--$lattice$.Insert($image$, $metadata$)
end for

Figure 4.5 – Algorithm for constructing an image lattice from an image database $\mathcal{I}$

on this bound do not achieve actual improvements in the implementations [26]. Furthermore, the number of nodes can become exponential in the number of property values.

Therefore, constructing an image lattice is an operation that can be done only off-line for large sets of images. The result is stored into the database. (Nevertheless, we can still envisage to use it on-line for presenting the few first results of a query.)

However, there exists an incremental algorithm. It consists in adding an image, based on its associated metadata, into an existing concept lattice. This happens to be very fast as long as new images do not create new nodes. Empirically, the complexity of adding one new node is in $O(n)$ where n is the number of nodes in the original lattice, whereas the complexity of adding a new image into an existing node is only in $O(\log n)$.

Therefore, the outer utilised algorithm is very simple (See Figure 4.5). In point of fact, it is exactly the translation of Function 4.13. Actually, the metadata consists of a set of strings in order to avoid a union type.

It is linear in the number of descriptors and the sizes of the images:

$$O(\sum_{j \in J_1} |\mathcal{D}_j| + |region| \times \sum_{j \in J_2} |\mathcal{D}_j| + \sum_{\forall i \in \mathcal{I}} width(i) \times height(i)).$$

With :

- $J_1 = \{area, orientation, elongation\}$
- $J_2 = \{black, white, darkgrey, lightgrey\} \cup (\{saturated, desaturated\} \times \{dark, light\} \times \{red, orange, yellow, green, cyan, blue, magenta\})$

Furthermore, as descriptors are bounded by constants in practise, the outer algorithm is actually linear in the sizes of the images.

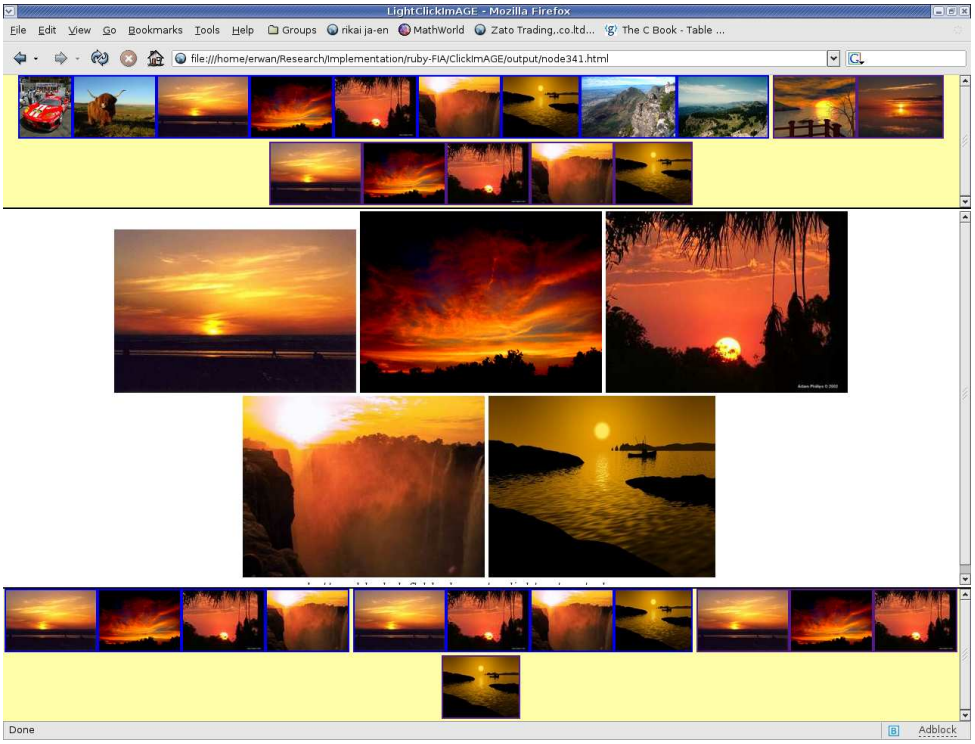

Figure 4.6 – The XHTML $Click^{Im}_{AGE}$ user's interface

We applied it to a small database of 3,000 images from a public domain collection, described later in section 4.4. Most of the images are located at the bottom of the lattice structure, whereas the upper nodes have no actual images, just an intension. Then, from the lattice, a set of XHTML pages have been generated.

Figure 4.6 shows a screenshot of browsing a navigation structure using a web browser. Each node is represented as a page divided into three parts delimited by the XHTML tag `div`. Top part contains a "tiny" representation of father nodes, central part a "standard" representation of current node, and bottom part a "tiny" representation of children nodes. The user does not need to describe *formally* hes query, he or she just has to click on images "he likes" in bottom part (if he or she wants a more specific query) or in top part (if he or she wants a more general query). In this way, naive user (who are not computer scientist) can easily get a set of images he or she likes.

4.3.1 Indexing a Galois Lattice in a RDBMS

In our first prototype, the user navigates through a set a generated XHTML pages. However, from these pages the structure can not be processed again to update the structure and add new images.

In order to keep this structure, we use a mapping of the structure in a database management system (DBMS). Although our implementation of the Galois lattice construction algorithm is object-oriented, our schema being quite simple a relational database management system

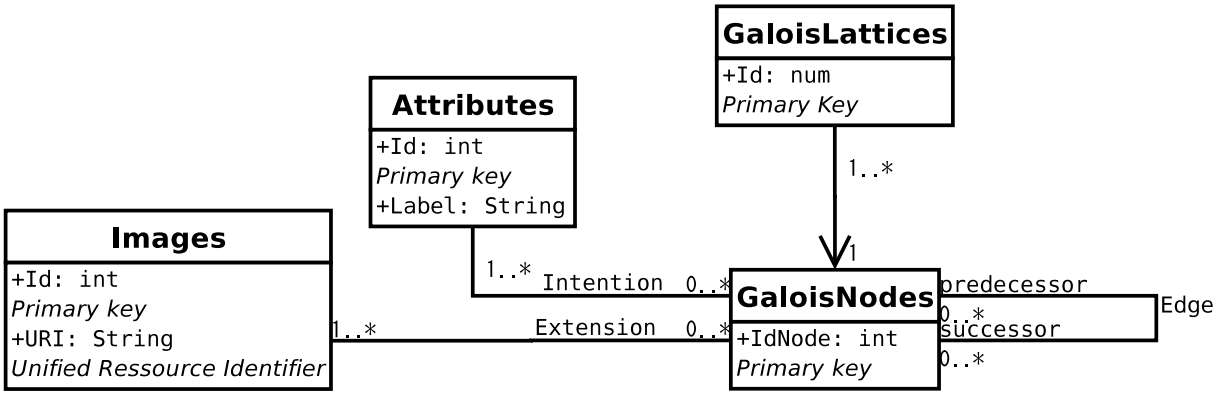

Figure 4.7 – The database schema

(RDBMS) fulfils our needs. Thus, we chose this model over object model to keep more control over the database.

4.3.2 Architecture

Figure 4.8 shows the architecture of our prototype. It consists in three modules: (1) GaloisLib, a library for creating and manipulating Galois' lattices written in C language for performance reasons. Lattices are constructed incrementally, using the algorithm proposed by [26]. The other modules are written in Ruby scripting language: (2) Metadata calculus module, based on the third-party library ImageMagick for image processing and (3) Output generation module, that generates the XHTML pages for navigation itself.

4.4 Experiments

We prepared a set of about 3,000 images, extracted from the Internet database *Flickr.com*. Flickr.com is a web site where anyone can upload their photographs to share them with their relatives, or anonymous visitors, and add key word annotations, so-called *tags*, such as "Tokyo", "wedding", or "dog." We randomly selected these 3,000 images using nine tags among the most popular ones (Flickr proposes such a list): *art*, *city*, *flower*, *party*, *sunsets*, *birthday*, *dog*, *snow*, and *nature*.

By using as a source a web site where anyone can post his or her own photographs, we are sure to use a *real world* data set. All the images are photographs that people actually take, and have been chosen because they represent a tag that is popular. Consequently, we consider our data set as being quite representative of the photographs that need to be indexed by an individual.

To select which descriptions to keep and which descriptions to discard, we set the α-cut to 0.3. For colour informations for example, that means that a given colour should cover at least 30% of one area to be kept in the metadata.

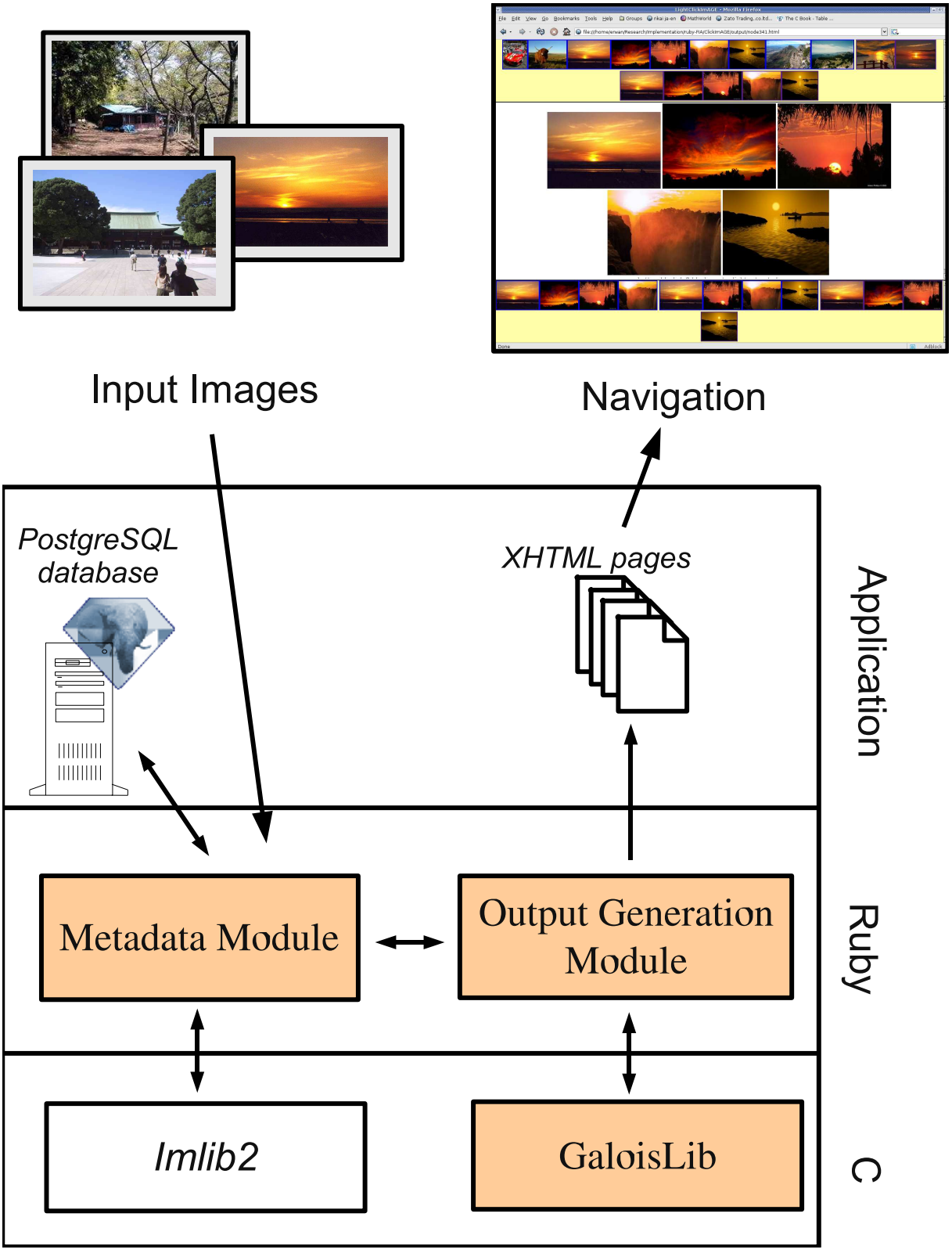

Figure 4.8 – The $Click^{Im}_{AGE}$ architecture

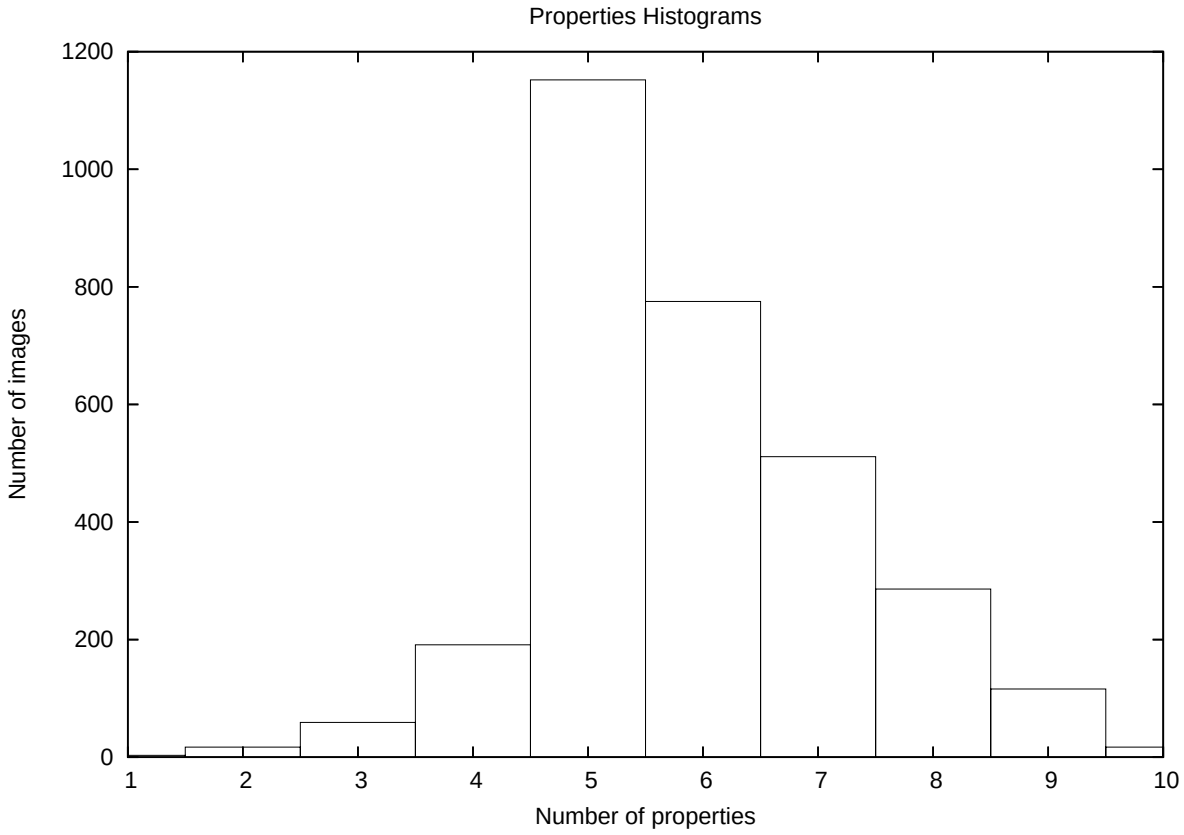

Figure 4.9 – Number of properties per image

4.4.1 Results

Figure 4.9 is calculated on the metadata of the collection, independently of the Galois' lattice. It shows how many properties an image features; we can see than using an α-cut of 0.3 on this collection all images are described between 2 and 10 labels, and most of the images have 5 labels.

That means that non-empty nodes (node having a non-null *reduced content*) will be in the first 10 levels; in other words, the information will be found in these 10 first levels.

Figure 4.4.1 shows the number of nodes in each level. 34 is the total number of properties; consequently, the superior node is in the 34^{th} level (all properties). Similarly, the inferior node is in the level 0 (no properties). The rest is strongly dependant on the average number of properties by images, illustrated by the histogram of figure 4.9.

The number of nodes in central levels (level 3, 4, 5) may look very high compared to the number of images indexed. However, one should note that nodes are connected to fathers and children; there is no connection between nodes of the same level. The number of connection for each node appears to be reasonable.

Level	**Nodes**	**Level**	**Nodes**
0	1	6	680
1	34	7	325
2	364	9	33
3	1041	8	113
4	1346	10	2
5	1097	34	1

CHAPTER 5

ADDITIONAL CLUSTERING

While navigation structures based on Galois' lattices offer several advantages, it has a a few disadvantages too. The most important problem is the scalability of this structure: the time complexity to build a Galois' lattice is in $O(n^2)$ [26], where n is the number of images. Generally speaking, for large data sets, the actual limit is known to be in $O(n.\log n)$ [71].

In this chapter, a solution to this scalability issue is proposed. We make use of a clustering method to divide the navigation process into two levels. The general level will be between homogeneous collections of images, and the specific level will be between images of the same collection. This proposal addresses the scalability problem, since the number of sub-collections will remain limited even on a very large set of data.

In this section, we propose to improve the navigation structure scalability by constructing the lattice not using *images* as the base element but *clusters of images*.

We propose to divide the navigation process into two levels. The general level is a navigation between *collections of images*. These collections are supposed to be different from each others and internally homogeneous. The specific level is inside a reduced set of images that are close to each others.

To build an appropriate set of images, we combine this first kind of classification technique with a clustering technique [60], the time complexity of which is only in $O(n)$. The net result is a *Galois' lattice of clusters*, the advantages of which are:

1. the image database is classified orthogonally along several properties, i.e., a kind of multidimensional indexing is achieved;
2. the speed at which the image database can be browsed and subsets of similar images retrieved is at most logarithmic in the number of features, i.e., *independent* of the number of images; and
3. the foreseen possibility to use this structure for physical indexing of large image databases.

In addition, the introduced browsing technique being based on a classification tool, it has the native property to group images into categories, i.e., a generalised form of the `group by` clause in querying languages.

First, in the section 5.1 we will present the SAINTETIQ system that is used to perform the classification technique. Then, in the section 5.2 we will detail the hypermedia representation

we use to present the *lattice of clusters* to the user and provide him (or her) an efficient user interface.

5.1 Details on the Clustering Method

The SAINTETIQ system [60] has been designed as a general database summarisation process, the summarisation of image databases as described below is one of its application. For the summarisation task, image instances are processed as database records with attributes being the image property and attribute values being the property descriptors of each instance, as defined in section 4.1.2.

Understandability and self-descriptive representation of summaries as well as database browsing ability are expected features of any summarisation process. Then, the symbolic/numerical interface provided by Zadeh's fuzzy set theory, and more especially linguistic variables and fuzzy partitions, are considered as fundamental theoretic background in most of the approaches to linguistic summarisation. Significant works have been done in this area, for instance by Yager [85], Bosc et al. [6], Cubero et al. [13], Dubois and Prade [15], Lee and Kim [36]

Considering all these approaches to database summarisation, SAINTETIQ benefits the same robustness and intelligibility of summary descriptions, as an application of fuzzy set theory. But in contrary to the approaches based on quantified statements [85] and gradual rules [6, 13], SAINTETIQ does not assume there exist input and output variables: our model is able to provide polythetic summaries, i.e. defined over all the attributes. Moreover, SAINTETIQ builds summaries with different granularity, rather than just considering a rewriting process into a predefined vocabulary as the Attribute Oriented Induction-based approach do [15]. Finally, the construction of summaries in SAINTETIQ is driven by data, in opposite to the naive approach with exhaustive generation of hypothetical summaries presented in [36].

The SAINTETIQ model considers a primary relation $\mathrm{R}(A_1, \ldots, A_n)$ in the relational database model, and constructs a new relation $\mathrm{R}^*(A_1, \ldots, A_n)$, in which tuples z are summaries and attribute values are fuzzy sets on linguistic labels describing a sub-part $\sigma_z(\mathrm{R})$ of R. Summaries z are then stored as fuzzy tuples into the relation R^*. Moreover, summaries are defined with different granulates, and organised into a hierarchy from the most general (the root) to the most specific (the leaves).

For instance, the summary z of R^*, represented on Figure 5.1 is defined as:

$$
\begin{aligned}
z.\texttt{CENTER} &= \{1.0/\textit{db. orange} + 0.8/\textit{db. violet} + \\
&\qquad 0.5/\textit{db. red}\} \ , \\
z.\texttt{TOP} &= \{1.0/\textit{vd. green}\} \ , \\
z.\texttt{BOTTOM} &= \{1.0/\textit{dd. violet} + 0.8/\textit{dd. green} + \\
&\qquad 0.5/\textit{black} + 0.8/\textit{vb. yellow}\} \ , \\
z.\texttt{LEFT} &= \{1.0/\textit{vd. violet} + 0.8/\textit{vd. red} + \\
&\qquad 0.5/\textit{vd. green}\} \ , \\
z.\texttt{RIGHT} &= \{1.0/\textit{vd. violet} + 0.8/\textit{dd. green} + \\
&\qquad 0.5/\textit{black}\} \ ,
\end{aligned}
$$

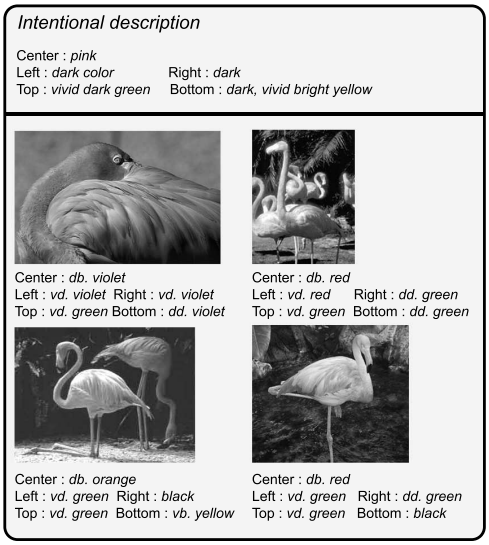

Figure 5.1 – Example of an image summary

where *vb.*, *vd.*, *db.*, *dd.* means respectively *vivid bright*, *vivid dark*, *dull bright*, *dull dark* nuances of the colour as explained in section 4.1. Membership grades of colour features are computed from the descriptions of the collection of images in $\sigma_z(\mathrm{R})$ as stated in [63].

5.1.1 Learning summaries from data

The SAINTETIQ system performs the database summarisation process by the way of a concept formation algorithm [17]. The process integrates learning and classification tasks, sorting each tuple through a summary hierarchy, and at the same time, updating summary descriptions and related measures.

Most of human learning can be regarded as a gradual process of concept formation : observation of a succession of objects allows to induce a conceptual hierarchy that summarises and organises human experience. In other words, concept formation is the fundamental activity which structures objects into a concise form of knowledge that can be efficiently used in the future. It includes the classification of new objects based on a subset of their properties (*the prediction ability*), as well as the qualitative understanding of those objects based on the generated knowledge (*the observation ability*).

The concept formation task is very similar to the *conceptual clustering* issue as defined by Michalski and Stepp [46], with the added constraint that learning is incremental.

More formally, given a sequential presentation of tuples and their associated descriptions, the main goals of concept formation are:

1. identifying clusters that group the tuples into categories;
2. defining an intentional description (i.e., a summary) that corresponds to each category;
3. organising these summaries into a hierarchy.

Incremental learning methods are basically dynamic: their input is a stream of objects that are assimilated one at a time. Thus, incremental processes build at any time an estimated knowledge structure of an unknown real one. Therefore, a primary motivation for using incremental systems is that knowledge may be rapidly updated with each new object. Indeed, incremental learners are driven by new objects, such that each step through the hypothesis space occurs in response to some new experience.

The fuzzy summary formation task is performed as a search through a space of summary hierarchies, and hill-climbing is a basic Artificial Intelligence search method providing a possible way of controlling that search. Indeed, the system adopts a top-down classification process, incorporating a new tuple t into the root of the hierarchy and descending the tree according to the hill-climbing search.

At a node z, the algorithm considers *incorporating* the current tuple t into each child node of z as well as *creating* a new child node accommodating t. Furthermore, the system evaluates the preference of *merging* the two best children nodes of z and *splitting* the best child node. Then SAINTETIQ uses a heuristic objective function [59] based on contrast and typicality of summary descriptions to determine the best operator to apply at each level of the hierarchy.

Furthermore, bidirectional operators, such as splitting and merging, make local modifications to the summary hierarchy. They are used to weaken sensitivity of the object ordering, simulating the effect of backtracking in the space of summary hierarchies, without storing previous hypotheses on the resulting structure. Thus, the system does not adopt a purely agglomerative or divisive approach, but rather uses both kind of operators for the construction of the tree.

To reduce effects of this well-known drawback of concept formation algorithms, one can consider an optimisation and simplification step, for instance with an iterative hierarchical redistribution [18] which considers the movement of a set of observations, represented by an existing cluster (summary), through the overall summary hierarchy, either by applying an additional iterative optimisation step [18], or by considering some extended bidirectional operators, as well as defining a new learning strategy [61]. The most interesting optimisation and simplification of hierarchical clustering seems to be the iterative hierarchical redistribution which considers the movement of a set of observations, represented by an existing cluster (summary), overall the summary hierarchy.

Finally, the main advantage of hill-climbing search is its low memory requirement, since there are never more than a few states in memory, by contrast to search-intensive methods as depth-first or breadth-first ones.

5.1.2 Selecting summaries for lattice generation

Building the lattice is a complex task and the number of nodes possibly incorporated within a reasonable amount of time is bound to a rather small value, depending on the computer memory and performance. For this reason, we will need to select a chosen number n of summaries for incorporation.

The extensive content of any of the hierarchy node is defined as the union of the extensive content of it's children nodes. Therefore, in order to avoid redundancy, only leaf nodes will be

incorporated in the lattice. Hence, selecting an upper level node means to discard all its lower level children.

The list of the leaf nodes is first build. The number of initial leaves is likely to be high as it reflects the number of description modalities found in the image database. The list is sorted according to the parent measures :

- *Extensive content cardinality*: the size of the effective content of the summary.
- *Similarity*: the extensive content homogeneity, where 1 means that all the summary content shares the same set of descriptors.

Those measures are defined in the SAINTETIQ process and updated during the summarisation task. From them, a comparison function is build which first orders summaries by decreasing cardinality and then, those with the same cardinality are ordered by increasing similarity.

While the number of remaining leaves is greater than the required number n, leaves with the lowest parent score are cut and all sibling nodes are replaced by their parent. Of course, this parent node is inserted in the ordered list according to its own parent ranking. As all the non-leaf nodes have at least two child nodes, this operation reduces the list at each step. At the end of this process, the list presents n or less summaries, ready to be incorporated into the lattice.

5.1.3 Building Galois' lattice of summaries

Each of the selected summaries has an intensional description where each attributes is associated with a fuzzy set of descriptors. As discussed in section 4.1.2, a Galois' lattice needs a crisp set of descriptors to be built. A threshold method ($\alpha - cut$) is used to select summary descriptors relevant enough to be used for the lattice generation. It is to be noted that a summary can possibly provide many descriptors for each attribute (image region), which is perfectly handled by $Click^{Im}_{AGE}$.

5.2 Hypermedia representation

5.2.1 Inter-clusters navigation

Representation of a Galois' lattice navigation structure in a hypermedia way depends on the *document* type. Representing an image is obvious since HTML proposes the `img` tag, but others documents may need investigation. To use Galois' lattice for a given document type, we have to be able to represent a document in two ways [40]:

- a small-sized displayable representation, used for the navigation task and
- a complete representation for investigation once the appropriate document is found.

The user interface displays, for any given position on the Galois' lattice, all the possible navigation directions as hyperlinks thanks to the first representation. Superior nodes (as defined in section 4.2) are displayed at the top of the screen and lower nodes at the bottom of it. The middle of the screen is used to display the actual content of the current node using the second representation.

In this application, documents are image summaries. The summary representation (feature descriptors) is used during the search process and the complete representation when the relevant document is found or in order to display the current node content.

5.2.2 Intra-clusters navigation

Each summary contains a subset of the images of the original image database. The limited size of the screen and the chosen thumbnail image size directly determines the maximum number of images N which will be displayed at the same time. In section 1.2, we stated that considering a 10^6 images database, each summary will contain an average of 200 images about a quarter of which will be displayable at the same time if we want to keep a reasonable size for the thumbnail images. Therefore, it is somehow interesting to provide users with the most relevant subset of images stored in a summary. This functionality provides the user with an overview of the summary content as accurate as possible at the first glimpse. The following of this section briefly describes how we take advantage of the built hierarchy structure to choose those samples [63]:

```
Choose(n integer, z summary) returns a set of images
   if (z content <= n)
     then return all z content
     else if (z is a leaf)
            then return n random samples
            else return for (each z_child) do
                          Choose((n / number_of_children(z)), z_child )
end of Choose;
```

In this function, parameter n refers to the number of images which are needed for immediate display.If the summary is a leaf, it reflects that images are very homogeneous according to the set of feature we used and that a random selection of them will provide a good idea of the summary content [74]. On the other hand, the children existence denotes that during the summary process, two different categories of images were ranked sufficiently different to justify this subdivision. Therefore, representative images of a summary should primarily be taken in each of its children to provide a representative sample of the summary content. Of course, the last part of this pseudo-algorithm is very simplified as it is necessary to deal with the different situations where the number of children of z is greater, lower or is not an integer divisor of n.

It is somehow interesting to provide users with a relevant subset of images stored in a summary. This functionality provides the user with an overview of the summary content as accurate as possible. If we want to avoid scroll bars, the limited screen size and the chosen size for thumbnail images directly determines the Maximum Number of Examples (MNE) that the program can display.

prototype images for summaries with child nodes

The children existence denotes that during the summary process, two different categories of images were ranked sufficiently different to justify this subdivision. Therefore, representative

images for a concept should primarily be taken in each of his children to provide a representative sample of the summary content. Child nodes are ordered according to their satisfaction degrees to the parent summary. Examples are taken one at a time, in this order, from each child summary until the MNE count is reached. If there are less child nodes than MNE, a second example could be taken from each child summary and this operation could be repeated until the MNE count is reached, or there is no more image. The way images are chosen inside child summaries is the same than the one used to choose images inside leaves and is described in the following.

prototype images for children or leaves

Images inside a leaf or inside child summaries are close enough to each other in regard of the current specialisation level. At that level, they are said to be homogeneous. Choosing representative images inside an homogeneous set has not much sense and that is why we propose three different ways to achieve the selection:

- First, images are ordered according to their satisfaction degrees with the summary description. This order provides a way to choose representative images. As shown earlier, this rank is based on the computed distance between linguistic description of images and summary intentions. If only one sample image is required, then the first of this list should be chosen. But, if more than one image is needed, we have to choose subsequent images in a way to maximise the between-image distance. Obviously, choosing images in their satisfaction order would exhibit very similar images, minoring the information carried by each sample.
- Secondly, the summary process restricts the features used to discriminate images to those with a referring vocabulary. In order to choose representative images of a summary in a leaf, other features or similarity measures may be used to discriminate images. In a leaf, the number of images is supposed to be a reasonably small subset of the all database content (since chosen features have to be relevant to discriminate images). Centroid methods may therefore be applied and more time consuming computation of image distances may be used to choose image samples.
- However, a random choice of the samples in the summary content could be as accurate as the above methods. According to Squire and Pun [74], random partition homogeneity is only 30% lower than human partition one. Thus, the more homogeneous is the summary, the more a random choice will be similar to more sophisticated methods.

CHAPTER 6

USER PERSONALISATION AND SUB-LATTICES

Content information is usually not enough to represent images as seen by a human observer. For example, one may see an image representing the Eiffel Tower as similar to an image representing the Arc de Triomphe (two monuments of Paris) while a system based only on content would rather see the Eiffel Tower image close to an image of Tokyo Tower (same shape).

However, rely on human annotation leads to several problems: not only manual annotation is very costly, but it remains a subjective annotation. Two annotators will produce a different annotation on the same image, even the same annotator would produce different annotations if he or she is asked to annotate the same image at different times.

Thus, we propose to offer user the possibility to define himself subjective annotations by applying masks on navigation structures. That means hide parts of the graphs or connections to display a subgraph closer to user's expectation, taking user's needs and specificities into account. This is done by keeping an underlying common structure to all users and all retrieval processes, consequently keeping the advantages of a before-hand calculated structure: the most costly processes are done before-hand, and retrieval itself is fast and reactive.

The problem of subjectivity becomes pointless since user establish the annotation for himself; there can be no distortion between the annotator and the user. Moreover, the cost of annotation is painless because it is integrated into the retrieval process.

6.1 Masking lattices

The time complexity of the Galois' lattice construction algorithm being experimentally $o(n^2)$ [26], it allows to reach a size of 10,000 instances [40]. In this case, a node explosion can happen and the path to the wanted image may be long. Moreover, if descriptions are randomly distributed on the image set, the number of edges can be very important and lead to confusion when user is to choose between too many children nodes.

In order to reduce the number of node by hiding only non-relevant one, and by limiting processing time, we propose to take the original Galois' lattice as a base to apply a mask.

A mask is a filter applied to a given Galois' lattice to hide elements, that can be nodes or links. It should be noted that while the resulting graph may not be a Galois' lattice since it will not represent a binary relation between two sets, it has to be a lattice (the axioms defining a lattice are presented in the section 4.2). Since the user will browse a direct representation of the resulting graph, every lattice axiom is mandatory to ensure that this browsing will allow the user to reach every non-masked image in a natural navigation path.

6.1.1 Formalisation

Different kinds of masking serve different goals. For example, one may want to reduce the cardinal of the images set or the cardinal of the description set. However, any kind of masking is represented in the same way.

Definition 11. *Given a lattice* $(\mathcal{N}, \mathcal{E})$*, a* lattice mask M *is defined as* $M = (N_M, E_M, E_A, N_{Me})$ *where* $N_M \subset \mathcal{N}$*,* $E_M \subset \mathcal{E}$*,* $E_A \subset \mathcal{N}^2$ *and* $N_{Me} \subset {N_M}^2$*. Also,* N_{Me} *is such as* $\forall (N_1, N_2) \in N_{Me}$*,* N_1 *is a father node of* N_2*.*

N_M represents the set of nodes to be masked, E_M the set of edges to be masked, E_A the set of edges to be added and N_{Me} the set of pair of nodes to be merged.

6.2 Masking techniques

In this section, we present one kind of filtering: *node masking*. It consists in masking some sets of images if the system already have informations about what kind of images are relevant to current retrieval and which images are not. Applying such a filtering will result in hiding complete nodes to user if most of its members are irrelevant to current search.

6.2.1 Node masking

The system operates a *node masking* when it has gathered informations about what kind of images user is looking for, enough to reduce the number of images to propose but not enough to give user a final result. Node identified as irrelevant to current retrieval should be masked.

A node masking operation is defined by a node filtering function f on nodes:

$$f : \mathcal{N} \rightarrow \{0, 1\}$$

The selection of nodes to mask is done by asking the user for examples of images to be masked, and inferring an approaching query. To ensure good performances, a low-complexity algorithm is chosen over better but high-complexity algorithms used in systems mainly based on relevance feedback.

Algorithm

Considering a node filtering function f and a Galois lattice $G = (\mathcal{N}, \mathcal{E})$, we note $N_F = N \in \mathcal{N} | f(N) = 0$ the set of nodes to mask. The following gives a algorithm to determine a mask $M = (N_M, E_M, F_M)$ that applied to G will result in a lattice according to section 4.2.1.

```
Nm <- Nf \ {min(G), max(G)};
FORALL n in Nm:
  FORALL e connecting n:
    add e to Em;
  FORALL p, parent node of Nf:
    CASE cardinal(non_masked_children(p)):
     0: FORALL c, child of Nf:
          add (p, c) to Ea;
     1: IF c, unique children of p
                     has no other parent:
          add (p, c) to Fm; add (p, c) to Em;
     else: nothing
  FORALL c, children node of Nf:
    CASE cardinal(non_masked_parent(c)):
     0: FORALL p, parent of Nf:
          add (p, c) to Ea;
     1: IF p, unique parent of c
                     has no other child:
          add (p, c) to Fm; add (p, c)to Em;
     else: nothing
```

Actually, this algorithm performs the following operations:

- The set of nodes to mask will be equal to the set of nodes defined by the filtering function, except that the minimum and maximum nodes cannot be masked,
- any edge connected to a masked node will be masked,
- if a node other than $min(G)$ ends with no parent, it should be connected to all parent of its last former parent
- if a node other than $max(G)$ ends with no child, it should be connected to all child of its last former child
- if a node ends with a unique child and this child has a unique parent, these nodes should be merged,
- if a node ends with a unique parent and this parent has a unique child, these nodes should be merged.

The complexity of this algorithm depends on the number of nodes to mask, and the average number of parents and children a node can have. Experimentally, we noticed that this number does not exceed a certain maximum. Indeed, since the low-level properties are correlated regarding their semantic meaning, we noticed that the number of children for a given node doesn't reach the number of properties but is at worst 25% it.

Thus, we conclude that this algorithm has an empiric linear complexity according the number of nodes to mask, i.e., in $O(n)$. This complexity is acceptable regarding the number of nodes to consider.

If a node had more than one parent, and all of them are masked in the process, then the result will depend on the last node masked by the algorithm. Since the order to process nodes is arbitrary chosen, this algorithm is not deterministic. However, parent nodes sharing all the same role, we do not see that point as a issue. There is a symmetric problem when masking children.

6.3 Conclusions

This masking technique provide a simple user-personalisation, allowing power-users to go further a simple browsing of the image collection.

It results in improving our previous proposal by adding user customisation without denying the performances advantages. It is consequently more efficient than a system based on feed-back querying or similarity search, and more relevant than a system based solely on a pre-calculated structure.

CHAPTER 7

FROM A FUZZY MODEL TO CRISP DESCRIPTIONS

We have seen in section 4.1 that a fuzzy model is adapted to describe images (based itself on numerical features), but to build a Galois' lattice we need a binary relationship: in a given image, a given property is either present or not. This is true not only for a Galois' lattice but also for any navigation structure that does not weight links between (groups of) images. Should the links be weighted, one should set a threshold to decide whether a neighbour should appear.

The most trivial answer, the one used in the original proposal presented in the chapter 4, is to rely on a constant threshold, determined empirically. However, doing so leads to the construction of sparse areas in the structure. In this chapter, we propose to limit the empiric space complexity by limiting these sparse areas as well as isolated elements.

Static or dynamic, the threshold has to be chosen in order to satisfy two conditions:

- adequacy to human perception: a human observer should most of the time agree with the system to see a property as "present" or not. This can be obtained by a form of training or just statistical analysis.
- building an efficient structure: the threshold should produce a number of properties per instance that is neither too small (it would lack precision), nor too big (this would hurt performances as well as avoid the system to discover similarities since each image would be too specific). Objective criteria exist, particularly the performance measure; however, subjective criteria such as the satisfaction of the user, or the ease of browsing are important but require exhaustive and costly experiments.

To achieve this goal, we present two techniques: *variable threshold* and *key matching insertion*.

Variable threshold takes into account global properties before applying a different threshold to each property. By doing so we wanted to increase the discriminative power of each property, however as detailed in the conclusion the results were not satisfying.

Key matching insertion is a different approach, that keeps the incremental nature of the construction algorithm. During the lattice construction, the algorithm is such as new images will be inserted in existing nodes when possible rather than to create new nodes. Note that when a new node is created for a new image, several nodes are actually created to ensure that the axioms

of Galois' lattices are respected; thus, nodes reduced to a single images have a great impact on the lattice size. The benefit of limiting the structure size is not only to save memory and disk, but also to provide a better navigation structure to the user. Indeed, a subgraph containing a great number of nodes for a small number of images would be disorienting for the user.

7.0.1 Variable Threshold

A key problem when trying to reduce noise (i.e., eliminating nodes that contain too few images with too few differences with respect to neighbouring nodes) while producing binary descriptors from fuzzy ones is that building a Galois' lattice is costly. Consequently, iterating the construction of successive Galois' lattices, even just two generations, each generation helping to remove noise for the next lattice is not a reasonable option! Therefore, for a given property, we search for a way to prevent it to form small nodes.

This process is based on two functions:

– A *presence* function: for a given property, it represents the presence of this property in the data set;
– A *mapping* function from the presence function to a threshold, to determine which threshold should be used for a given (image, property) couple regarding its isolation degree.

7.0.1.1 Introducing a presence function

We declare a presence function as follows:

$$f : \mathcal{D} \to [\mathit{1}, \infty) \tag{7.1}$$

for a given Galois' lattice $R = \mathcal{I} \times \mathcal{D}$.

As a first approximation, we propose the following isolation degree computation f_1, named *simple presence*, applied to lattice R:

$$f_1 : \begin{array}{l} \mathcal{D} \to [0, \infty) \\ d \mapsto \frac{\sum_{i \in \mathcal{I}} \mu_d(i)}{|\mathcal{I}|} \end{array} \tag{7.2}$$

where $\mu_d : \mathcal{I} \to [0, 1]$ represents the membership degree of a given image in the fuzzy subset associated to description d.

For a given property, this function computes the sum of the membership degrees for all the images; thus it represents the importance of this property in the set.

With adequate data structures, the cost of this pre-processing step is only in $O(|R|)$, i.e., optimal.

7.0.1.2 Finding an appropriate mapping function

Next, the mapping function should provide a threshold for a given isolation degree. Basically, the more the element is isolated, the more it is likely to be noise and a severe threshold should be applied. We expect the mapping function f to have the following properties:

1. $g(0) = T_{\max}$ where $T_{\max}$ is a constant experimentally fixed.
2. $\lim_{x\to\infty} f = T_{\min}$ where $T_{\min}$ is a constant experimentally fixed.
3. $g'(x) < 0$: the function should obviously be decreasing to have isolated elements to be filtered by a greater threshold.
4. $g''(x) > 0$: in order to be perceptually linear, the threshold should be decreasing faster for small values of x (number of neighbours). Thus the second derivative should be positive.

A logarithmic function is commonly used to translate human perception, but (1) it does not have an $Y = constant$ line as an asymptote, and (2) it is decreasing too fast for our needs: to keep the function in a $[T_{\min}, T_{\max}]$, we must apply a very factor so strong that it makes the curve look like the one of a linear growth function. Thus, we choose to base our proposal on the inverse function.

The simplest function that fits our needs is the following[1]:

$$g : \begin{array}{l} \mathbb{R} \to \mathbb{R} \\ x \mapsto \frac{1}{Ax+B} + C \end{array} \tag{7.3}$$

where A, B and C are positive constants. Applying the conditions $f(0) = T_{\max}$ and $\lim_{x\to\infty} f = T_{\min}$, we obtain:

$$g : \begin{array}{l} \mathbb{R} \to \mathbb{R} \\ x \mapsto T_{\min} + \frac{1}{Ax + \frac{1}{T_{\max}-T_{\min}}} \end{array}$$

It comes that the derivative function $g'(x)$ is:

$$g'(x) = \frac{-A}{(Ax + \frac{1}{T_{\max}-T_{\min}})^2}$$

which is actually negative for all positive values of x, so f is a decreasing function. Then, the second derivative is:

$$g''(x) = \frac{2A^2}{(Ax + \frac{1}{T_{\max}-T_{\min}})^3}$$

which is actually positive, ensuring that the derivative function is increasing, thus the absolute value of the derivative function is decreasing.

7.0.2 Key Matching Fuzzy Insertion

The alternative described above is rather complex since we had to find adequate functions. In this section, we propose another technique to build a Galois' lattice from a fuzzy relationship. Here, we shall not convert the fuzzy descriptors into crisp ones but rather build incrementally the lattice and select which descriptions to use *according to the existing lattice*. The goal of this procedure is still to avoid noise and build a lattice with compact nodes in order to get an

[1] For the domain set is $\mathbb{N}$, we should define our mapping function from $\mathbb{N} \to \mathbb{R}$. However, in order to define properties on the derivative and the second derivative functions we work directly on $\mathbb{R} \to \mathbb{R}$.

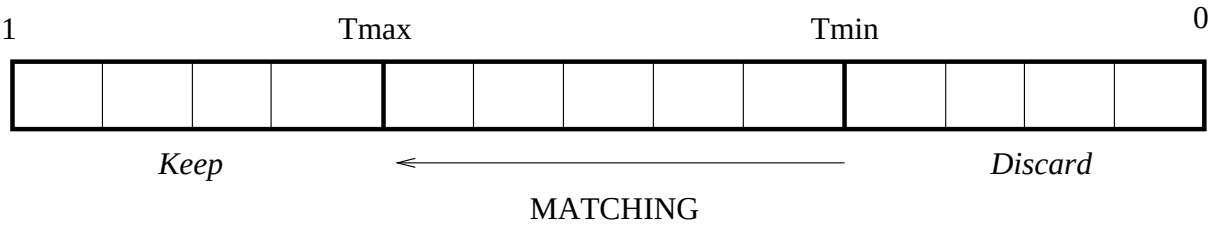

Figure 7.1 – Key Matching

efficient navigation structure. The difference with the previous approach is that, at each step, the actual structure and the content of the lattice help in deciding the way to manage the next image insertion. Therefore, the variable threshold is determined dynamically for each image and each descriptor.

Using a simple threshold technique, we noticed that some parts of the lattice were very sparse. When images only share a few properties, the result can be at worse an exponential explosion of the number of nodes, *i.e.* each descriptions combination results in the creation of a new node. The resulting structure is obviously bad for navigation, since (1) the user will have to specify (implicitly) each description to enter and (2) the number of children of a given node will be very big, thus the user's choice will become harder.

Using this technique, we intend to force the creation of meta-descriptions (a set of description appearing together in a lot of nodes) and thus create a better quality structure.

Hereafter, we call "key" the set of descriptions associated to a given image. The following algorithm is used in order to determine which descriptions should be used when inserting a given image into an existing lattice:

- Descriptions are ordered from the description having the highest membership degree to the description having the lowest membership degree (see Figure 7.1);
- Descriptions under a $T_{\min}$ threshold are discarded and descriptions over a $T_{\max}$ threshold are kept;
- Starting from the set of descriptions over $T_{\max}$, we add successively the descriptions in the interval $[T_{\max}, T_{\min}]$ and try to match the longest resulting key with existing keys in the lattice;
- When a matching key is found, this key is used to insert the image;
- If no matching key are found, the image is inserted with the set of descriptions over the threshold $\frac{T_{\min}+T_{\max}}{2}$.

Since the number of properties for a given image is barely constant, this algorithms has a linear complexity, $O(n)$, where n is the number of nodes of the existing lattice.

The benefit of this algorithm is as follows: the goal being to reduce the number of nodes for better retrieval, this algorithm ensures that a given image is inserted, whenever possible, into an existing node. In fact, the standard lattice construction algorithm appears to be very efficient, i.e., logarithmic when inserting an element, the key of which already exists. Therefore, this modified insertion method promises to be more efficient.

7.1 Evaluation and Experiments

In order to determine which of is these two empirical approaches is actually interesting, we have to conduct some experiments and compare them, using the naive constant threshold approach as the baseline. On the basis of the code that was written to test Galois' lattice-based navigation, we conducted such experiments (1) to determine if the proposed approaches give results interesting enough from the computational cost point of view and (2) to determine which parameters give the best results from the noise reduction point of view.

Based on the same code that was written to test Galois' lattices for the basic navigation process, we conducted experiments:

1. to determine if our extension actually gives the expected results, and
2. to ensure that there is no increase of the time complexity. We used the image collection detailed in section 4.4.

On this set, as well as on subsets, a Galois' lattice is constructed for different values of the parameters. Several metrics are calculated on these lattices in order to evaluate the quality of the lattice in term of usability. Results of the *variable threshold* technique are based on previous experiments performed on a base of 1,700 images.

7.1.0.1 Metrics

We define the following metrics on a lattice:

– Cardinal: the number of nodes of the lattice; since information is hard to find in a too large structure, we prefer a lattice with a limited number of nodes.
– Average size of nodes: the size of a node is defined as the number of images in its *reduced content*. The reduced content of a node is defined in section 4.2.3; we use it rather than extension in order to match the representation of the nodes from the user's point of view, also described in the same section.
– Ratio of *real* nodes: we call *virtual node* a node, the *reduced content* of which is reduced to the empty set. Non-virtual nodes are called real nodes. This metric shows the proportions of nodes actually containing data in the lattice, the virtual nodes being only useful for navigating.

We made measures on the variable threshold as well as on key matching fuzzy insertion. We then compared them to the trivial technique, based on a constant threshold. For constant insertion, we chose a threshold of $T = 0.3$. For key matching insertion, we chose $T_{\min} = 0.2$ and $T_{\max} = 0.4$. These values were determined empirically, trying to match human perception, and we chose on purpose the threshold of constant insertion as the average of $T_{\min}$ and $T_{\max}$. In this way, we ensure than the difference observed is not due to a difference in the threshold choice.

7.1.1 Results

Figure 7.2 and 7.3 show respectively the average node size and standard deviation on the node size for different values of the factor A, as well as the curve for a constant threshold. These

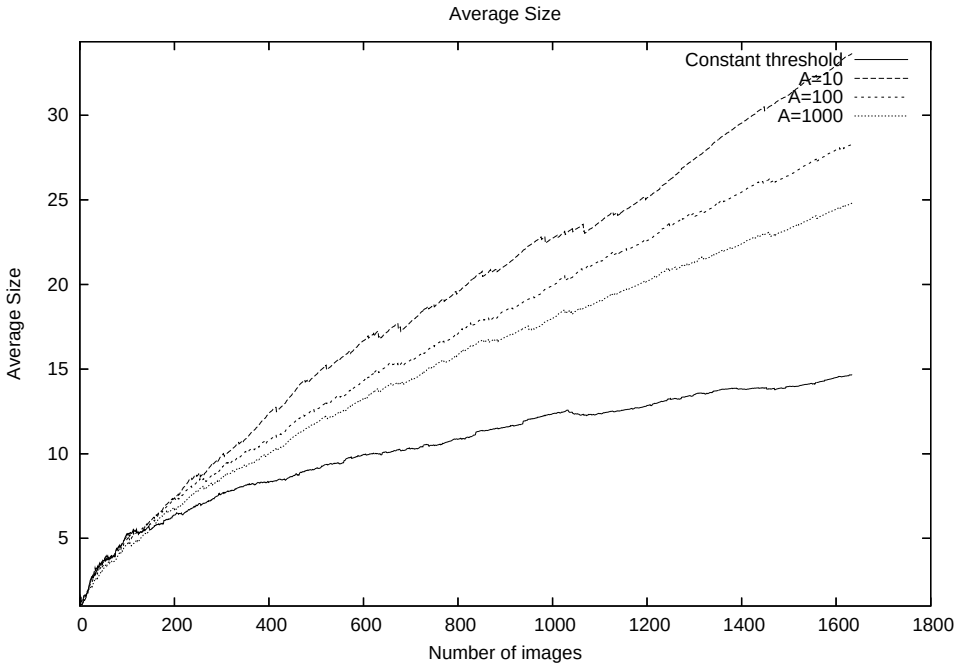

Figure 7.2 – Average node size, for different values of A for variable thresholding

curves are for $T_{\min} = 0.2$, $T_{\max} = 0.4$, and the isolation function used is the simple isolation function f_1.

On these curves, we cannot see a clear advantage in favour of the variable threshold-based technique using the *simple isolation* function. The curves are very close to what we observed on curves of constant threshold for different value of the threshold T.

We analyse these poor results as being a side-effect that we did not expect beforehand: most images featured a similar number of properties, and consequently the relevant subpart of the lattice is even more reduced.

However, key matching fuzzy insertion demonstrates a clear difference with respect to constant threshold. Figure 7.4 shows the percentage of real nodes, i.e., nodes that actually contain information, while other nodes are only present for a navigational purpose. Of course, with only one image, the lattice is reduced to a unique node that contains this image and thus is not virtual. Consequently, when the number of images is 1, the ratio is 1; however, in order to have a better representation of other parts of the graph, we chose a scale that stops at $y = 0.5$.

Figures 7.5 and 7.6 respectively show the total number of nodes and the average size of a node. The total number of nodes is clearly lower for key matching insertion; in this sense the structure will be easier to browse than the big structure built by the trivial algorithm. Having less nodes, the lattice built by key matching obviously has bigger nodes. However, the size of each node is still small enough for a comfortable browsing.

When looking at figure 7.6, we notice that there are several local extrema on both curves. Moreover, local extrema on the lower curve, representing the constant threshold approach, do not correspond to local extrema for the key matching upper curve. For example, around 700,

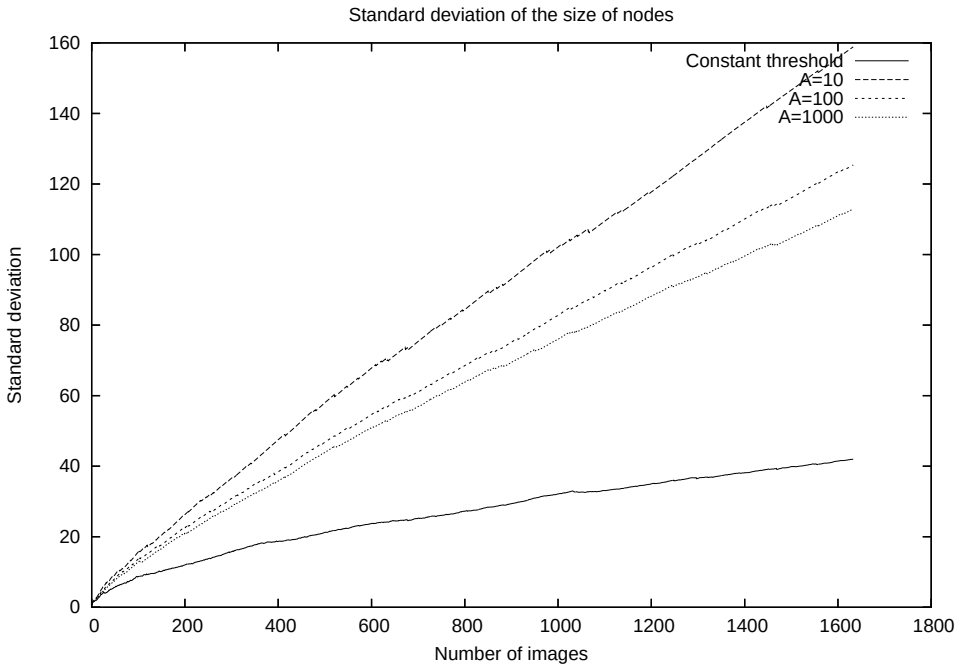

Figure 7.3 – Standard deviation on node sizes

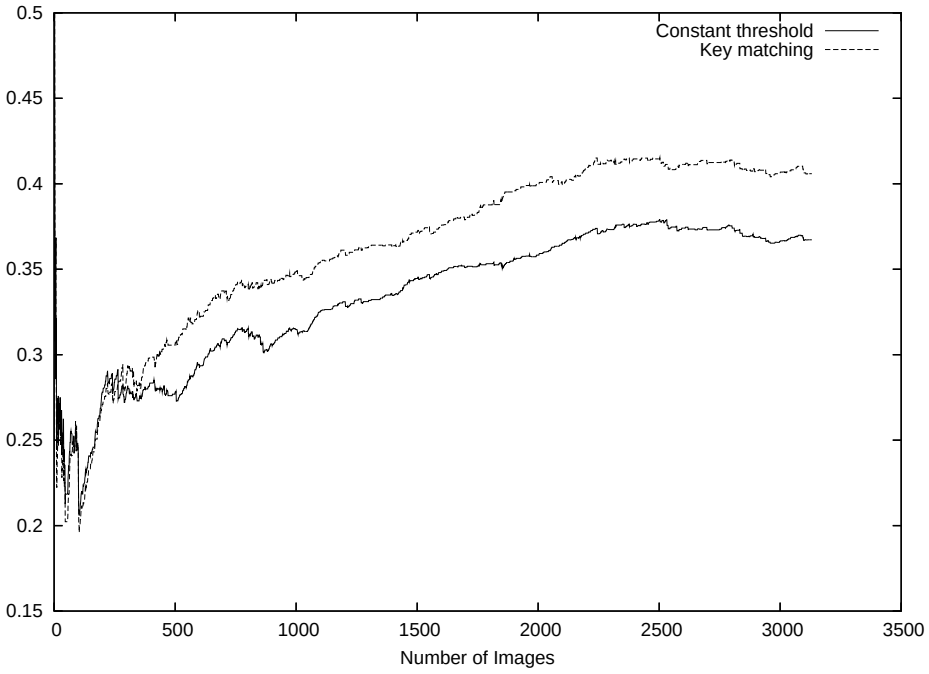

Figure 7.4 – Ratio of Real Nodes in the Lattice

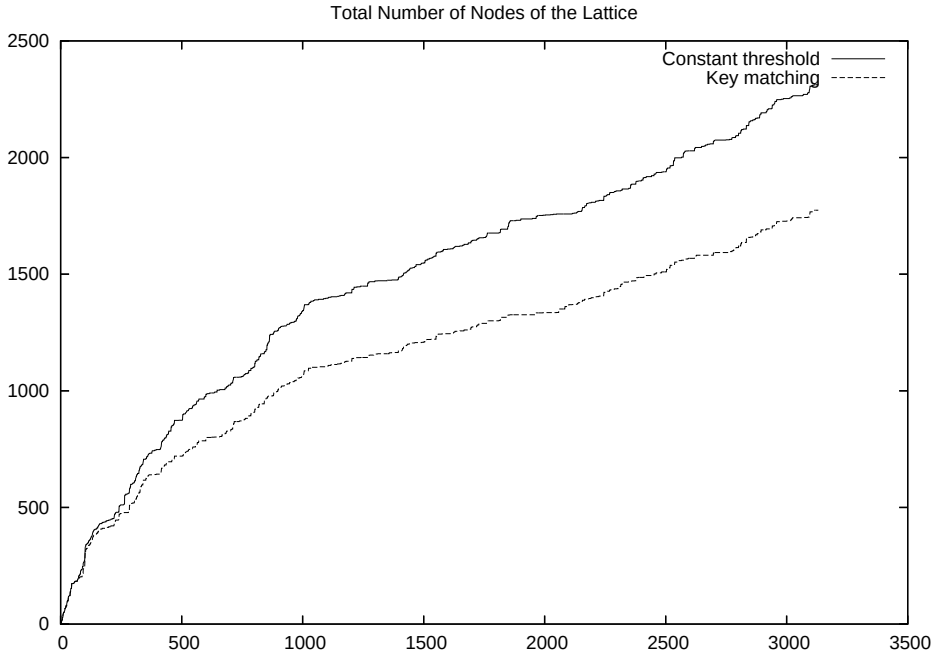

Figure 7.5 – Total Number of Nodes

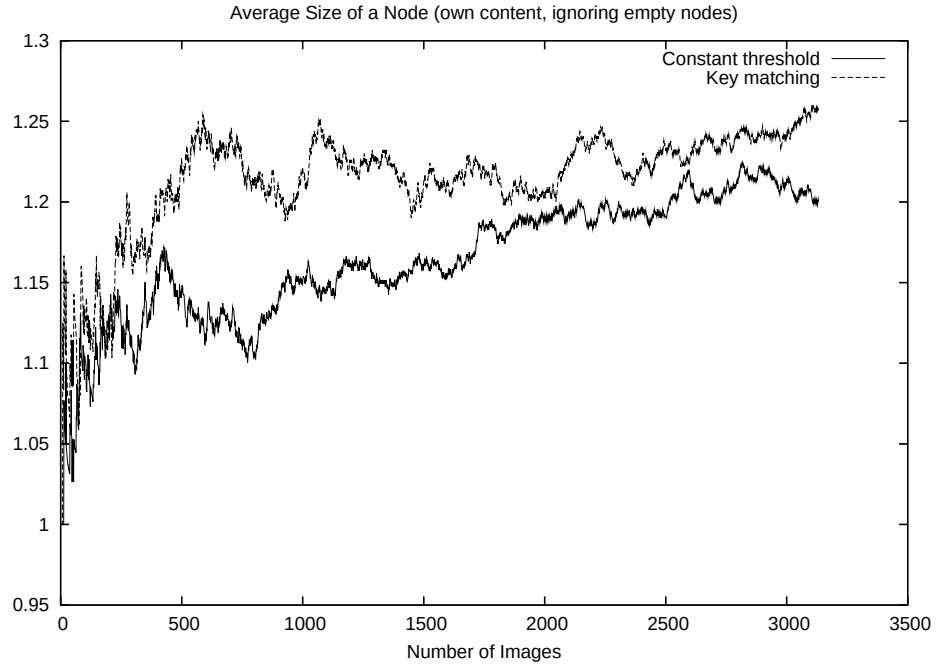

Figure 7.6 – Average Size of a Node (excluding virtual nodes)

there is a local maximum for the key matching approach while the curve for constant threshold is still decreasing. In fact, after the 693rd image, the insertion of a new category starts: images tagged with *flower* (after *art* and *city*). Photographs related to *flower* are very specific: the border is green representing leaves or grass, and the centre is colourful representing the flower itself. Consequently, new descriptions and new combinations appear, forcing the key matching algorithm to create new nodes reduced to one element: the average size of a node decreases. Shortly after, when enough nodes have been created, the average size of a node increases again. Before the 693rd image, when images where quite similar, the size of each node was increasing for the key matching algorithm whilst the constant threshold algorithm failed to see similarities in the descriptions.

Unfortunately, user's experience is not as easily quantifiable. Currently, our personal experience as well as the comments of a few volunteers confirm the statistical results: retrieving images from a lattice built with key matching is easier than with constant threshold. The reason is mainly that the number of nodes has been reduced, in particular there are fewer virtual nodes. Thus, browsing from one part of the lattice to another is both faster and more informative.

7.2 Conclusions

In this section, we presented two techniques to build a Galois' lattice from a fuzzy relationship. The second method we presented improved the resulting structure quality compared to the trivial method (applying a constant threshold to each description in each image).

By *improving quality* we mean that the number of small nodes have been reduced; such nodes create zones that require a lot of user interaction for a small amount of information. Moreover, this improvement is done without impacting the algorithmic complexity of the lattice construction algorithm.

While it has been tested on an image lattice, it is important to note that this technique can be used to build a Galois lattice from any fuzzy relationship. In particular, since the clustering technique presented in the chapter 5 produces fuzzy descriptors of each cluster, a *lattice of clusters* can be build using the proposal presented in this chapter.

CHAPTER 8

CONCLUSIONS

In this study, we looked for a way to replace query by navigation to search for images in an image collection or a database. Our work resulted in a technique based on Galois' lattices, a graph structure that shown to be useful for both indexing and retrieval by grouping images sharing common properties.

While lacking precision for users who need detailed queries, it is an easy-to-use yet powerful search for users who prefer to browse fastly an important set of images. This Galois' lattices-based technique could be applied to other media type, but it is particularly adapted to images since it is a still media that can be visualised quickly even in a reduced format.

On the one hand an image search based on navigation through Galois' lattice had several advantages:

- Navigation is very fast,
- a Galois' lattice is intrinsically a multi-dimensional classification technique,
- the tool is insensitive to correlations,
- it helps to correct users' mistakes very easily,
- the Galois' lattice structure easily hides unwanted features.

But there were also notable drawbacks, the biggest problem being the lack of scalability. We addressed the scalability problem by combining the Galois' lattice to a clustering technique; then we improved the structure quality by taking global properties into account, without increasing the algorithmic complexity of the lattice construction. Finally, we introduced a user personalisation process to take into account the different needs of users.

Compared to other techniques based on navigation, the specificity of our approach is to be solely based on a before-hand calculated structure, launching no query. Consequently, it is extremely fast and responsive, allowing user to go and back through the structure without having to wait for a query result processing.

The main application we thought for our proposal is a structure to build a static collection, for example the catalogue of an image provider. However, the construction algorithm being incremental it can be used for a dynamic collection, since the collection does not exceed a certain size (a few thousands of images using the basic proposal).

8.1 Benefits and Limits to Our Proposal

Since most content-based image retrieval systems (CBIR) are based on separate models for the indexing and retrieval processes, they cannot be easily compared to our proposal based on a single graph structure both for indexing and retrieving.

From a user point of view, our approach has the following advantages:

– First of all, we benefit of advantages specific to Galois' lattices, detailed in the section 1.2. Its main advantages are that navigation is very fast, easy even when the user is not able to describe his needs and is insensitive to correlations. Despite these advantages, the current proposal is the first to make use of Galois' lattice directly as a navigation structure.
– Since the main problem of Galois' lattices, *scalability*, has been addressed by using a complementary clustering technique, our proposal can easily reach one million images.

However, its major weak points is that the scalability problem has been solved by adding a new navigation level, in order to build the lattice on a smaller number of elements. The complexity still remains the same. While building a navigation structure over a very large collection (one million images) is now possible, this does not mean that the system is actually scalable, i.e. of linear complexity. A linear augmentation of resource does not permit a linear augmentation of the number of images.

That means that our system is not adapted to open collections such as the world wide web. We don't think that any system based on Galois' lattices can be useful for such application; it should be applied on quite stable collections such as professional images providers catalogues, or individuals' photograph collections.

8.2 Further Work

While we explored deeply the use of Galois' lattices for navigation through image collections, there are still work directions to be explored.

8.2.1 Applications to Other Media Type

In this study, we focused on *images*. However, we think that Galois' lattices may also be interesting for other kind of visual media, such as videos.

One member of the Nantes' University BADRI research team has already started to apply this work to video. As well as any other kind of media, two things are required to build a navigation structure based on Galois' lattices.

– A suitable metamodel, *i.e.* a set of descriptions that can be associated to any element along with a membership degree (a real number of the interval $[0, 1]$.)
– A compact way to represent it, in order to show to user the content of a node. It should be possible to display between 10 and 20 elements on the same screen.

Additionally, if the Galois' lattice is combined with a clustering process as described in the chapter 5, define an adapted way to navigate inside a node (*i.e.* navigate through a small collection of elements - between 10 and 100 - independently of the lattice) can be necessary.

Applying the current proposal to media type such as audio, that are not visual, would raise more problems. Unless the audio extract is mainly speech that can be converted to text, the user needs to *listen* to the extract in order to decide whether it is relevant or not for his or her needs. Obviously, the user can only listen to one extract at time. That makes a big different with image retrieval, where the user can see from 10 to 20 images at a glance.

8.2.2 Mobile Computing

In this scope, an interesting application is *mobile computing*. Indeed, recent mobiles phones are equipped with a digital camera. In some country like Japan, digital camera for mobile phone became a common accessory, and recently even cheapest devices are equipped with a high-resolution camera.

Using these devices, users quickly take a lot of photographs that are stored on the device itself or a memory card. Due to the limitation of input devices on mobile phones, adding keywords and organising images is a hard task; on such devices, an automatic organisation of images would be an advanced in the possibility of user to access to his own data.

We believe that Galois' lattices could be a good way to organise a user's images on his mobile phone equipped with a digital camera. For this application, our proposal has the following advantages:

- Since our approach is based solely on content information, inserting a new image requires no user interaction. This is precious since typing on a 9-keys mobile phone's keyboard is quite painful.
- A mobile phone can provide additional information, mainly localisation and date. Since Galois' lattices is insensible to correlation, we can just add these information as new metadata to take part of them.
- The user interaction of our system is already very simple: the user navigates only by clicks. This is much easier to adapt to a mobile phone keyboard than an interface that would be based on queries, or even feedback querying.

The main issue to be solved is the small screen of these devices. Even with screen resolution increasing (recent models have a 240×320 pixels resolution, and this number is likely to increase), the useful size of the screen will still be limited by the size of the device itself; the device have to fit in the hand of users. Consequently, compared to a desktop computer far less images can be displayed simultaneously on the screen.

List of Figures

Part I — State of the Art

Part II — Efficient Structures for Navigating an Image Collection

Bibliography

[1] S. ABITEBOUL. Object databases support for digital libraries. In SPRINGER-VERLAG, réd., *Proceedings of the 1st European Conference on Research and Advanced Technology for Digital Libraries (ECDL'97)*, number 1324 in Lecture Notes in Computer Science (LNCS), pages 39–45, Pisa, Italy, September 1-3 1997.

[2] J. F. ALLEN. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843, September (November XXX) 1983.

[3] D. H. BALLARD et C. M. BROWN. *Computer Vision*. Prentice-Hall, 1982. 523 p.,

[4] F. BARBEAU et J. MARTINEZ. About tours in the OTHY hypermedia design. In SPRINGER-VERLAG, réd., *Proceedings of the 5th International Computer Science Conference: Internet Applications (ICSC'99)*, number 1749 in Lecture Notes in Computer Science (LNCS), pages 146–155, Hong Kong, China, December 13-15 1999.

[5] E. BERTINO, B. C. OOI, R. Sacks-Davis K.-L. TAN, J. ZOBEL, B. SHIDLOVSKY et B. CATANIA. *Indexing Techniques for Advanced Database Systems*. Kluwer Academic, Boston, Massachussets, 1997. 250 p.,

[6] P. BOSC, O. PIVERT et L. UGHETTO. On data summaries based on gradual rules. In *Proc. of the Int. Conf. on Computational Intelligence, 6th Dortmund Fuzzy Days (DFD'99)*, volume 1625 of *LNCS*, pages 512–521, Dortmund, Germany, may 25-28 1999. Springer.

[7] M. BOUET, A. KENCHAF et H. BRIAND. Shape representation for image retrieval. In *Proceedings of the 7th ACM International Multimedia Conference (ACM-MM'99)*, volume 2, pages 1–4, Orlando, Florida, November 1999.

[8] C. CARSON et V. E. OGLE. Storage and retrieval of feature data for a very large online image collection. In *IEEE Computer Society Bulletin of the Technical Committee on Data Engineering*, volume 19 of *4*, pages 19–27, December 1996.

[9] G.-H. CHA et C.-W. CHUNG. Object-oriented retrieval mechanism for semistructured image collections. In *Proceedings of the 6th ACM International Multimedia Conference (ACM-MM'98)*, Bristol, UK, September 14-16 1998.

[10] S-F. CHANG, J.R. SMITH, M. BEIGI et A. BENITEZ. Visual information retrieval from large distributed online repositories. *Communications of the ACM*, 40(12):63–67, 1997.

[11] M. CORRIDONI, A. Del BIMBO et P. PALA. Image retrieval by color semantics. *Multimedia Systems*, 7(3):175–183, 1999. Springer-Verlag,

[12] I. J. COX, M. L. MILLER, S. M. OMOHUNDRO et P. N. YIANILOS. PicHunter: Bayesian relevance reedback for image retrieval. In *Proceedings of 13th International Conference on Pattern Recognition (ICPR'96)*, pages 361–369, Vienna, Austria, 1996.

[13] J. C. CUBERO, J. M. MEDINA, O. PONS et M.-A. VILA. Data summarization in relational databases through fuzzy dependencies. *Information Sciences*, 121(3-4):233–270, 1999.

[14] J.-P. DELAHAYE. La ressemblance mathématisée. In *Pour la science*, number 235, pages 361–369. May 1983.

[15] D. DUBOIS et H. PRADE. Fuzzy sets in data summaries - outline of a new approach. In *Proc. of the 8th Int. Conf. on Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU'2000)*, volume 2, pages 1035–1040, Madrid, July 3-7 2000.

[16] C. FALOUTSOS, R. BARBER, M. FLICKNER, J. HAFNER, W. NIBLACK et D. PETKOVIC. Efficient and effective querying by image content. *Journal of Intelligent Information Systems*, 3(3-4):231–262, 1994.

[17] D. H. FISHER. Knowledge acquisition via incremental conceptual clustering. *Machine Learning*, 2:139–172, 1987.

[18] D. H. FISHER. Iterative optimization and simplification of hierarchical clusterings. *Artificial Intelligence Research*, 4:147–179, avril 1996.

[19] M. FLICKNER, H. SAWHNERY, W. NIBLACK, J. ASHLEY, Q. HUANG, B. DOM, M. GORKANI, J. HAFNER, D. LEE, D. PETKOVIC, D. STEELE et P. YANKER. Query by image and video content: The QBIC system. *IEEE Computer*, 28(9):23–32, September 1995.

[20] W. B. FRAKES et R. BÄEZA-YATES, réds. *Information Retrieval: Data Structures & Algorithms*. Prentice-Hall, 1992. 504 p.,

[21] B. GANTER. Two basic algorithms in concept analysis. *Technische Hochschule Darmstadt*, 1984.

[22] M. R. GAREY et D. S. JOHNSON. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, New York, 1979.

[23] F. GARZOTTO et P. Paolini D. SCHWABE. HDM – a model-based approach to hypertext application design. *ACM Transaction on Information Systems*, 11(1):1–26, January 1993.

[24] T. GEVERS. *Colour Image Invariant Segmentation and Retrieval*. Thèse de Doctorat, University of Amsterdam, The Netherlands, May 1996. 142 p.,

[25] T. GEVERS et A. SMEULDERS. A comparative study of several color models for color image invariant retrieval. In *Proceedings of the 1st Internationale Workshop on Image Databases & Multimedia Search*, Amsterdam, The Netherlands, 1996.

[26] R. GODIN, R. MISSAOUI et H. ALAOUI. Incremental concept formation algorithms based on Galois (concept) lattices. *Computational Intelligence*, 11(2):246–267, 1995.

[27] J. HAFNER, H. SAWHNERY, W. EQUITZ, M. FLICKNER et W. NIBLACK. Efficient color histogram indexing for quadratic form distance functions. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 17(7):729–736, July 1995.

[28] W. HSU, T. S. CHUA et H. K. PUNG. An integrated color-spatial approach to content-based image retrieval. In *Proceedings of the 3rd ACM International Multimedia Conference (ACM-MM'95)*, pages 303–313, 1995.

[29] J. HUANG, S. R. KUMAR et R. ZABIH. An automatic classification scheme. In *Proceedings of the 6th ACM International Multimedia Conference (ACM-MM'98)*, Bristol, UK, September 14-16 1998.

[30] T. ISAKOWITZ, E. STOHR et P. BALASUBRAMANIAN. RMM: A methodology for structured hypermedia design. *Communications of the ACM*, 38(8):34–44, August 1995.

[31] B. JÄHNE. *Digital Image Processing: Concepts, Algorithms, and Scientific Applications*. Springer-Verlag, 1991. 402 p.,

[32] J. S. JIN et R. KURNIAWATI. A scheme for intelligent image retrieval in multimedia databases. *Journal of Visual Communication and Image Representation*, 7(4):369–377, December 1996.

[33] G. JOMIER, M. MANOUVRIER et M. RUKOZ. Stockage et gestion d'images par un arbre quaternaire générique. In *Acte des 15èmes Journées Bases de Données Avancées (BDA'99)*, volume October, pages 405–424, Bordeaux, France, 1999.

[34] T. KÄSTER, M. PFEIFFER, C. BAUCKHAGE et G. SAGERER. Combining Speech and Haptics for Intuitive and Efficient Navigation through Image Databases. In *Proc. International Conference on Multimodal Interfaces (ICMI'03)*, pages 180–187. ACM, 2003.

[35] P. M. KELLY, T. M. CANNON et D. R. HUSH. Query by image example: The CANDID approach. In *SPIE Storage and Retrieval for Image and Video Databases III*, volume 2420, pages 238–248, 1995.

[36] D. H. LEE et M. H. KIM. Database summarization using fuzzy isa hierarchies. *Ieee Trans. On Systems Man & Cybernetics-Part B: Cybernetics*, 27:68–78, feb 1997.

[37] G. LEVY et F. BAKLOUTI. Parallel algorithms for general galois lattices building. In *Workshop WDAS.*, 2003.

[38] W. Y. MA et B. S. MANJUNATH. NETRA: A toolbox for navigating large image databases. In *Proceedings of the IEEE International Conference on Image Processing (ICIP'97)*, 1997.

[39] J. MARTINEZ et S. GUILLAUME. Colour image retrieval fitted to classical querying. *Networking and Information Systems Journal (NISJ)*, 1(2-3):251–278, 1998.

[40] J. MARTINEZ et E. LOISANT. Browsing image databases with Galois' lattices. In *Proceedings of the ACM International Symposium on Applied Computing (SAC'02)*, pages 971–975, Madrid, Spain, March 11-14 2002. ACM Computer Press.

[41] J. MARTINEZ et S. MARCHAND. Towards intelligent retrieval in image databases. In B. BERRA, réd., *Proceedings of the 5th IEEE International Workshop on Multi-Media*

Data Base Management Systems (IW-MMDBMS'98), pages 38–45, Dayton, Ohio, August 5-7 1998. IEEE Computer Press.

[42] José MARTINEZ. An hypermedia framework for visualising databases contents. In Noureddine MOUADDIB, réd., *Actes des 17e Journées Bases de Données Avancées (BDA)*, pages 187–195, Agadir, Morocco, octobre 2001. Cépaduès Éditions.

[43] M. MECHKOUR. EMIR2: An extended model for image representation and retrieval. In *Proceedings of the 6th International Conference on Database and Expert Systems Applications (DEXA'95)*, London, UK, September 4-8 1995.

[44] C. MEGHINI. An image retrieval model based on classical logic. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'95)*, pages 300–308, Seattle, Washington, July 1995.

[45] C. MEGHINI, F. RABATTI et C. THANOS. Conceptual modelling of multimedia documents. *IEEE Computer*, 24(10):23–32, October 1991.

[46] R. S. MICHALSKI et R. E. STEPP. Learning from observation: Conceptual clustering. In Ryszard S. MICHALSKI, Jaime G. CARBONELL et eds TOM M. MITCHELL, réds., *Machine Learning, an Artificial Intelligence Approach*, pages 331–363. Tioga Publishing Co., Palo Alto, CA, 1983.

[47] F. NACK et A. LINDSAY. Everything you wanted to know about MPEG–7: Part 1. *IEEE Multimedia*, 6(3):65–77, July/September 1999.

[48] F. NACK et A. LINDSAY. Everything you wanted to know about MPEG–7: Part 2. *IEEE Multimedia*, 6(4):64–73, October/November 1999.

[49] C. NASTAR, M. MITSCHKE et C. MEILHAC. Efficient query refinement for image retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'98)*, Santa Barbara, California, June 23-25 1998.

[50] C. NASTAR, M. MITSCHKE, C. MEILHAC et N. BOUJEMAA. SurfImage: A flexible content-based image retrieval system. In *Proceedings of the 6th ACM International Multimedia Conference (ACM-MM'98)*, Bristol, UK, September 14-16 1998. ACM, ACM Press.

[51] J. NIELSEN. *Hypertext and Hypermedia*. Academic Press, San Diego, California, 1990. 268 p.,

[52] V. E. ÖGLE et M. STONEBRAKER. Chabot: Retrieval from a relational database of images. *IEEE Computer*, 28(9):40–48, September 1995.

[53] M. ORTEGA, Y. RUI, K. CHAKRABARTI, S. MEHROTRA et T. S. HUANG. Supporting similarity queries in MARS. In *Proceedings of the 5th ACM International Multimedia Conference (ACM-MM'97)*, Seattle, Washington, November 1997.

[54] G. PASS et R. ZABIH. Comparing images using joint histograms. *Multimedia Systems*, 7(3):234–240, 1999. Springer-Verlag,

[55] A. PENTLAND, R. W. PICARD et S. SCLAROFF. PhotoBook: Content-based manipulation of image databases. *International Journal of Computer Vision*, 18(3):233–254, 1995.

[56] R. W. PICARD. A society of models for video and image libraries. *IBM Systems Journal*, 35(3-4), 1996.

[57] R. W. PICARD et T. P. MINKA. Vision texture for annotation. *Multimedia Systems*, 3(1):3–14, February 1995.

[58] R. W. PICARD, T. P. MINKA et M. SZUMMER. Modeling subjectivity in image libraries. In *Proceedings of the IEEE International Conference on Image*, Lausanne, Switzerland, September 1996.

[59] G. RASCHIA et N. MOUADDIB. A fuzzy-based heuristic measure evaluating quality of a concept partition: Application to SaintEtiQ, a database summarization system. In *Proceedings of the 9th IEEE International Conference on Fuzzy Systems (FUZZ-IEEE'2000)*, volume 2, pages 957–960, San Antonio, Texas, May 7-10 2000. (short paper),

[60] G. RASCHIA et N. MOUADDIB. SaintEtiQ: A fuzzy set-based approach to database summarization. *International Journal of Fuzzy Sets and Systems*, 129(2):137–162, July 2002.

[61] J. ROURE et L. TALAVERA. Robust incremental clustering with bad instance orderings: A new strategy. In Helder COELHO, réd., *Progree in Artificial Intelligence - IBERAMIA 98, 6th Ibero-American Conference on AI*, volume 1484 of *LNAI*, pages 136–147. Springer, Lisbon, Portugal, Oct 1998.

[62] E. SABER, A. M. TEKALP, R. ESCHBACH et K. KNOX. Automatic image annotation using adaptative color classification. *Graphical Models and Image Processing*, 58(2):115–126, March 1996.

[63] R. SAINT-PAUL, G. RASCHIA et N. MOUADDIB. Prototyping and browsing image databases using linguistic summaries. In *Proc. of the IEEE Int. Conf. on Fuzzy Systems (FUZZ-IEEE'2002)*, Honolulu (Hawaii), USA, May 2002.

[64] G. SALTON et M. M. GILL. *Introduction to Modern Information Retrieval*. MacGraw Hill, New York, 1983.

[65] H. SAMET. The quadtree and related hierarchical data structures. *ACM Computing Surveys*, 16(2):188–260, June 1984.

[66] S. SANTINI et R. JAIN. Similarity is a geometer. *Multimedia Tools and Applications*, 5(3):377–406, November 1997.

[67] S. SANTINI et R. JAIN. Beyond query by example. In *Proceedings of the 6th ACM International Multimedia Conference (ACM-MM'98)*, Bristol, UK, September 14-16 1998.

[68] Simone SANTINI, Amarnath GUPTA et Ramesh JAIN. Emergent semantics through interaction in image databases. *IEEE Transactions on Knowledge and Data Engineering*, 13(3):337–351, 2001.

[69] Simone SANTINI et Ramesh JAIN. Integrated browsing and querying for image databases. *IEEE MultiMedia*, 7(3):26–39, – 2000.

[70] G. SHEIKHOLESLAMI, W. CHANG et A. ZHANG. Semantic clustering and querying on heterogeneous features for visual data. In *Proceedings of the 6th ACM International Multimedia Conference (ACM-MM'98)*, Bristol, UK, September 14-16 1998.

[71] S. SKIENA. *The Algorithm Design Manual*. ISBN 0-387-94860-0. Springer & Verlag, 1997.

[72] A. W. M. SMEULDERS, M. WORRING, S. SANTINI, A. GUPTA et R. JAIN. Content-based image retrieval: the end of the early years. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2000.

[73] J. R. SMITH et S.-F. CHANG. VisualSEEK: A fully automated content-based image retrieval system. In *Proceedings of the 4th ACM International Multimedia Conference (ACM-MM'96)*, pages 87–98, Boston, Massachussets, November 1996.

[74] D. SQUIRE et T. PUN. A comparison of human and machine assessments of image similarity for the organization of image databases, 1997.

[75] M. STRICKER et A. DIMAI. Color indexing with weak spatial constraints. In *Storage and Retrieval for Image and Video Databases IV, SPIE Proceedings Series*, volume 2670, pages 29–40, February 1996.

[76] M. STRICKER et M. ORENGO. Similarity of color images. In *Storage and Retrieval for Image and Video Databases III, SPIE Proceedings Series*, volume 2420, pages 381–392, 1995.

[77] M. J. SWAIN et D. H. BALLARD. Color indexing. *International Journal of Computer Vision*, 7(1):11–32, 1991.

[78] M. SZUMMER et R. W. PICARD. Indoor-outdoor image classification. *Proceedings of the International Workshop on Content-Based Access of Image and Video Databases (CAIVD'98)*, 1998. IEEE Computer Society,

[79] Y. TAO et I. GROSKY. Spatial color indexing: A novel approach for content-based image retrieval. In *Proceedings of the IEEE International Conference on Multimedia Computing and Systems (ICMCS'99)*, pages 530–535, Firenze, Italy, June 1999.

[80] A.M. TURING. Computing Machinery and Intelligence. *Mind*, 59(236):433–460, 1950.

[81] M. VAZIRGIANNIS. Uncertainty handling in spatial relationships. In *Proceedings of the ACM Symposium on Applied Computing (SAC'2000)*, volume 1, pages 494–500, Como, Italy, March 19-21 2000. ACM Press.

[82] H. WANG, F. GUO et D. D. FENG. A signature for content-based image retrieval using a geometrical transform. In *Proceedings of the 6th ACM International Multimedia Conference (ACM-MM'98)*, Bristol, UK, September 14-16 1998.

[83] M. E. J. WOOD, N. W. CAMPBELL et B. T. THOMAS. Iterative refinement by relevance feedback in content-based digital image retrieval. In *Proceedings of the 6th ACM International Multimedia Conference (ACM-MM'98)*, Bristol, UK, September 14-16 1998.

[84] J. K. WU, B. M. Mehtre A. D. NARASIMHALU, C. P. LAM et Y. J. GAO. CORE: A content-based retrieval engine for multimedia information systems. *Multimedia Systems*, 3(1):25–41, February 1995.

[85] R. R. YAGER. A new approach to the summarization of data. *Information Sciences*, 28(1):69–86, octobre 1982.

Browsing a Classification of an Image Collection

Erwan LOISANT

Mots-clés : base de données, images, recherche d'information, navigation, treillis de Galois

Keywords: image, database, information retrieval, navigation, Galois' lattice